Decomposition And Identification of Molecules

Peter Kolb
kolb.peter@gmail.com

August 2, 2016

Contents

1 Purpose of DAIM

Our approach to docking is to cut ligands into pieces, dock these independently of each other with the program SEED [1, 2] and then use this information to place the entire ligand correctly with FFLD [3]. To make the docking (and the decomposition) of large libraries feasible, DAIM was developed.

Shortly said, the tedious work of looking at the screen, counting atoms and typing the obtained information in an input file is now replaced by a couple of command lines. DAIM is capable of identifying the most promising fragments in a molecule and computing their properties. As the fragments docked by SEED should be minimised, DAIM can also compare the found fragments to already existing and fully prepared molecules. One additional feature is the possibility to filter databases for the occurrence of certain fragments in the molecules. Furthermore, whole databases can be analysed to obtain data on rotatable bond counts, donors/acceptors a.s.o.

This manual is split into two main parts:

- one that lists the most common problems you will meet in the docking business and provides step-by-step instructions on how to run DAIM in those cases (chapters 3 & 4). This is where you should start reading. However, already there you will find references to the second part to omit redundancies.
- a second part that provides a complete manual of all commands and possibilities, as well as descriptions of the underlying concepts which would in principle allow you to develop this program further (chapters 5–11).

2 Using and Referencing DAIM

Before running DAIM, you should obtain a license. This license can be requested online at `http://www.biochem-caflisch.uzh.ch/download/`. When citing DAIM, please refer to:

P. KOLB AND A. CAFLISCH
Automatic and efficient decomposition of two-dimensional structures of small molecules for fragment-based high-throughput docking.
J. Med. Chem., **2006**, *49*, 7384-7392.

Invoking DAIM for the first time the user will be prompted to specify the location of ausxiliary files. By typing the absolute path to the location of the underlying DAIM distribution DAIM will copy all necessary files into `~/.daim`.

3 Main Uses of DAIM

Before you begin, you are strongly encouraged to (1) take a look at `daim.param` (in `cwd` [current] or `~/.daim` [default]) and see if all the settings make sense for your project and (2) run DAIM with the `--`check option on all the molecules to spot potential problems.

A **Preparing 10,000 (or any other number ...) molecules for a SEED/FFLD run – but no minimised library available from previous runs.**

1. First you have to make a choice: for SEED you need a definition of the binding site.
   (A) If you have the structure of a cocrystallisation available, you can pass both the receptor and the ligand to DAIM, which will compute the necessary information
   ↪ use options `-r` (`--`receptor) *and* `--`ligand.
   (B) If you can not – or do not want to do that, you have to give a list of the numbers of the residues forming the binding site to the program
   ↪ use options `-r` (`--`receptor) *and* `--`bsfile.
   For information on both options please refer to chapter 8.2.
2. In the directory that contains the library of molecules type

   ```
   ls -1 *.mol2 > foo
   ```

`foo` should have 10,000 lines.

3. In case **(B)**, create a file that contains a list of the sequence numbers of the residues in the binding site, each number on a new line.
4. Case **(A)**: run DAIM with

   ```
   daim_x -s -r receptor.mol2 --ligand native_ligand.mol2 -c 10000 --list foo
   ```

   Case **(B)**: run DAIM with

   ```
   daim_x -s --bsfile bind.site -r receptor.mol2 -c 10000 --list foo
   ```

   In `daim.log` you'll find progress information, you can check this file during and after the run.
5. After the run: In case **(A)**, the input file for SEED will be complete. For case **(B)** (i. e. if you used the `--`bsfile option), it will not contain points in the binding site (used for the angle cutoff) and also no cutoff sphere. This information would have to be added by hand (for more information on the SEED input-file cf. 9.6 and [1]).

   In the directory specified by the parameter `SEED_MOL2_DIR` (default `./inputs/`, cf. 7.3.4) you'll find the fragments DAIM identified. Each fragment is unique within the limitations of the fingerprints (cf. 11). These you have to minimise and assign partial charges to and then you can start your SEED calculation.

B **Filtering the library of 10000 for molecules containing certain fragments and/or those that are similar to known ligands.**

1. Obtain the fingerprint(s) (cf. 5.1) of the fragment(s) and/or the molecule you want to filter for: run DAIM on a molecule that contains this fragment (using an empty database). You might want to set *Frag.Threshold* in `daim.param` to 0 or use '`-`w all' to ensure that all fragments are listed.

   ```
   daim_x --html -m molecule.mol2
   ```

   Load all the fragments in `./inputs/` into a molecule viewer and identify your fragment. Note the number and look for the fingerprint in the output file. The fingerprint of the whole ligand is printed close to the top of the file.
2. Additionally (or alternatively) you can use the Lipinski [4] or Veber [5] rules or the `.mol2` file of a substructure as criterion.
3. Set up the filter file. Please read chapters 7.6 and 5.4 on how to do that.
4. Run DAIM (if you want a SEED input file, include the `-`s, `-`r and maybe also the `--`bsfile options)

   ```
   daim_x -f filter.file -c 10000 --list foo
   ```
5. Check the result. DAIM will tell you which of the molecules were skipped because of the filter in `daim.log` (you can grep for 'filter' there). `daim_all.dat` (cf. 9.8.1) will contain only the molecules that passed the filter.

C **Obtain information on the structural diversity of the library.**

1. There are two ways in DAIM with which you can gather information: `--`qsar, which provides an overview of the occurring fragments and `--`cluster, which groups the molecules according to their fingerprints.
2. Execute step 2 of problem A.
3. Run DAIM on this library, this time giving the `--`qsar and/or the `--`cluster option

   ```
   daim_x --html --qsar [--cluster] -c 10000 --list foo
   ```
4. In the outfile there will be links to two output files. `daim.stats` contains a list of the fragments and how often each of these appeared in the molecules of the database. `daim.clus` features the data on the clusters and which molecule can be considered as "cluster representative", i.e., the molecule lying closest to the cluster center.

D **Generate a database of fingerprints from 300 minimised fragments.**

1. In the directory containing the minimised fragments type

   ```
   ls -1 *.mol2 > foo
   ```

2. Run DAIM

   ```
   daim_x -d 300 --list foo
   ```

   The only output will be a file called `daim.db` containing the fingerprints. Be careful not to overwrite existing databases! If you want to call this library in a different way, use `--`dbfile [filename].

4 FAQ

- *I wanted to recompile* DAIM*, but it doesn't work.*
  When I run DAIM *it can't find the libcgicc.so.5.0.1 file.*

  DAIM uses the cgicc-classes, which are linked dynamically to the executable. When you want to recompile or DAIM complains, you have to install these files on your computer. You can find them on the web at `http://www.gnu.org/software/cgicc/cgicc.html`. DAIM was built using version 3.2.3 of this library.

- *I've installed the cgicc-class, but it still doesn't work!*

  Make sure that you have installed the cgicc-class as su. Verify furthermore that `/usr/local/lib` (the default installation pathway) is in `/etc/ld.so.conf` . Add it if necessary and, in any case, run `ldconfig` (maybe twice).

- *There is no input file for SEED or it is not complete.*

  Did you set the -s option? Did you then also specify a filename for the receptor (-r option)? Is this name of the receptor a single filename (good) or does it contain a full path (can lead to mistakes)?

- *I'm not satisfied with the way* DAIM *decomposes a molecule. I would like it to consider fragment F in molecule X. Can I set somewhere that* DAIM *cuts at a specific bond?*

  You can set new or erase existing additional non-rotatable or unbreakable bonds in `daim.param` and also modify their characteristics and definitions (cf. 7.3.3). Furthermore, you can hinder the choice of fragments by passing their fingerprints to DAIM with `--`exclude (8.2). However, you *cannot* force DAIM to *cut* a certain bond *B*. If you really want to decompose a single molecule in a specific way, you probably have to do it by hand.

- *Can I directly use the fragments* DAIM *writes out for a SEED-run?*

  In principle yes. They fulfil all the requirements to a `.mol2`-file. However, these fragments will clearly *not* be minimised and they will also *not* have the correct partial charges assigned (cf. 11). This means that SEED will complete the run, but that you should carefully check the results.

- *Where can I obtain the fingerprints for the fragments I want to screen for?*

  There are two ways: either you take a molecule that contains this fragment and run DAIM on it. Then you might have to use the -w option to find out which one the desired fragment is. Alternatively, you can create the fragment with a drawing program and let DAIM find the fingerprint for you. The fingerprint can be found in the similarity analysis section of the output HTML file.

  In any case it is *not* a good idea to use the -a or -d options, since the molecules will not be decomposed that way and you can't be sure that DAIM would not cut any bonds...

- *I added a number of non-rotatable bonds in the parameterfile, but* DAIM *still treats some of them as rotatable. Why?*

  Make sure that all the atom types in the `.mol2`-file are correct, and that they really are CHARMM atom types !

5 Theoretical Concepts

5.1 The Fingerprints

To characterise the fragments and molecules, *fingerprints* are used. These contain information about several chemical features (Table 1).

field no.	0	1	2	3	4	5
meaning	NumAtoms	NumC	NumN	NumO	NumHal	NumS
	6	7	8	9	10	11
	NumP	NumAromaticBnd	NumDoubleBnd	NumTripleBnd	NumAmideBnd	NumAcce
	12	13	14	15	16	17
	NumDonor	NumRings	totRingSize	longestChain	Wiener Index 4/1000	Num2N
	18	19	20	21	22	
	Num3N	Num4N	NumHA	NumRotBnd	NumRingBnd	

Table 1: Entries into Fingerprints

As a sample, the fingerprint for aniline (Fig. 1) is shown.

```
aniline  14 6 1 0 0 0 0 6 0 0 0 0 2 1 6 7 0.54 0 7 0 7 0 6
```

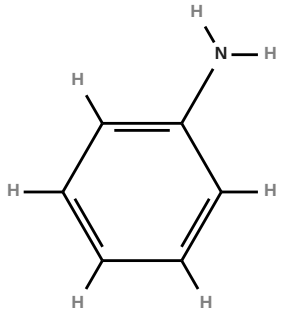

Figure 1: Aniline.

The first number (field 0[1]) states that in total there are 14 atoms in this molecule, 6 of which are carbons and 1 is a nitrogen (fields 1 and 2, respectively). Looking at fields 3 to 6 we know that there are no other atoms – except hydrogens – present. Field 7 tells us that there are 6 aromatic bonds, the following 0 discloses the absence of any double bond. In field 11 the number of H-bond acceptors is given whereas the next field contains the number of H-bond donor directions. The 14^{th} field states the number of rings (given by the cyclomatic number [6]: $n_{bonds} - n_{atoms} + 1$), the following number is the total number of heavy atoms in rings and the next one denotes the longest chain of atoms in the molecule. Field 16 contains the Wiener Index 4, scaled by a factor of 1000 (cf. 5.1.3 and [7]). Fields 17, 18 and 19 state the number of atoms having two, three or four neighbors, respectively. Field 20 gives the total number of heavy atoms, field 21 the number of rotatable bonds and field 22 number of bonds in rings. For a review on similarity matching and fingerprint types see [8].

5.1.1 H-Bond Donors and Acceptors

An H-atom is treated as a possible H-bond direction if it is connected to either O, N or S. Oxygen is considered to be an acceptor by default. For a nitrogen, the number of neighbours is considered: if it has one or two neighbours, it is an acceptor. In case it has three neighbours and at least one of that neighbours is sp^2-hybridised, it is not considered as an acceptor, otherwise it is. And should it have four neighbours, it is also not considered as H-bond acceptor. For sulfur, only thioketones are considered to be acceptors.

A list of the atoms that are donors or acceptors can be obtained by passing the option `--hbo` (cf. 8.2 and 9.4).

5.1.2 Longest Chain

This property is defined as the length of the longest sequence of atoms, including atoms in rings and hydrogens. It is determined by carrying out a "breadth first" search. Please bear in mind that the algorithm will always take the shortest path through a ring.

[1] This part of the manual is also for developers. Since counting in C++ always starts at 0, I have kept this numeration.

5.1.3 Wiener Index 4

The Wiener Index can be regarded as a descriptor of the topology of molecules. Originally, it was defined as [9]

$$W = \frac{1}{2}\sum_{i,j=1}^{n} s_{ij} \tag{1}$$

where s_{ij} is the ij-entry in the distance matrix of the molecular graph. Clearly, this does not take into account the properties of the atoms connected by the bonds.

Yang *et al.* developed a new definition for the index, which makes use of both the distance matrix and two vectors representing atom properties [7]. It can be written as

$$W_F = \boldsymbol{XWL} = (x_1, x_2, \ldots, x_n)\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} \tag{2}$$

In the DAIM implementation, the β_i and x_i are defined as in equations (3) and (4), respectively.

$$\beta_i = (I_i)^2 \tag{3}$$

$$x_i = \frac{1}{p_i}\sum_{j=1}^{p_i} X_{ij}, \text{ where } X_{ij} = \frac{1 + \Delta I_{ij}}{R_{ij}} \tag{4}$$

I_i is Pauling's electronegativity, p_i the number of neighbours of atom i, $\Delta I_{ij} = |I_i - I_j|$, the absolute value of the difference in electronegativity between atoms i and j and R_{ij} the sum of the covalent radii of atoms i and j, respectively.

5.1.4 Bertz Index

The Bertz Index is a measure of the complexity C of a molecule and was suggested by Bertz in the early 1980's [10]. Later on, Hendrickson *et al.* simplified the formula and adapted it to individual atoms [11]. It consists of two main parts:

$$C = C_\eta + C_E \tag{5}$$

C_η is the major term and measures the skeletal complexity, as defined in eq. (6).

$$C_\eta = 2\eta\ log_2\ \eta - \sum_i \eta_i\ log_2\ \eta_i \tag{6}$$

The individual η_i are calculated as

$$\eta_i = {}^1\!/_2(4 - h)(3 - h) \tag{7}$$

where h is the number of hydrogens on atom i. η for the whole molecule can be computed with equation (8), where h_i is the number of hydrogens on atom i, D the number of double bonds and T the number of triple bonds in the molecule.

$$\eta = \frac{1}{2}\sum_i (4 - h_i)(3 - h_i) - D - 3T \tag{8}$$

C_E (equation (9)) is called the symmetry term and is a function of the diversity of the elements, E being the total number of non-hydrogen atoms and E_j the number of atoms of element j.

$$C_E = E\ log_2\ E - \sum_j E_j\ log_2\ E_j \tag{9}$$

DAIM will compute the Bertz Index when the option `--bertz` is set (cf. 8.2).

5.1.5 CLogP

Since the fingerprints are a record of chemical and structural features, they can be combined to estimate molecular descriptors, such as the logP (octanol/water partition coefficient), which can be calculated by atom-additive methods [12, 13]. DAIM uses the – rather simple – formula $CLogP = a_1 \cdot fp_1 + a_2 \cdot fp_2 + \cdots + a_n \cdot fp_n$. The coefficients $\{a_1, a_2, \ldots, a_n\}$ have been derived by a fit of the fingerprints of the 1863 molecules from the library provided by Wang *et al.* together with the program XLOGP v2.0 [14] to their *measured* LogP values using the adaptive Particle Swarm Optimiser (cf. 11). Since they do not significantly contribute to the LogP of the training set molecules, fields 5, 6 and 17 (NumS, NumP and Wiener 4 Index) are included neither in the fit nor in the actual computation.

5.2 Similarity

Fingerprints can be compared with several measures. DAIM supports three: the Tanimoto coefficient (a similarity coefficient), the Euclidian metric (a distance metric) and the cosine coefficient (again a similarity coefficient). All measures will be able to find the (same) perfect match. If such a match does not exist, the suggested best match can differ. The list of the 10 best matches will mostly be the same, differing only in the ranking of some matches.

The criterion for a "good match" is the parameter *Cutoff_for_Similarity* (cf. 7.3.1). If the best value of *sim* is smaller (using the Tanimoto or cosine coefficient) or larger (using Euclidian metric) than this value, the fragment is written to a file for further processing.

5.2.1 Tanimoto Coefficient

The similarity coefficient is given by

$$sim = \frac{\sum_{i=0}^{n} x_{iA} x_{iB}}{\sum_{i=0}^{n} x_{iA}^2 + \sum_{i=0}^{n} x_{iB}^2 - \sum_{i=0}^{n} x_{iA} x_{iB}} \tag{10}$$

with x_{iA} and x_{iB} being the values at a certain position in fingerprint A and fingerprint B, respectively. This way, *sim* is between 0 and 1 (identity).

5.2.2 Euclidian Metric

Here, the distance value can be calculated with

$$sim = \sqrt{\sum_{i=0}^{n} (x_{iA} - x_{iB})^2} \tag{11}$$

x_{iA} and x_{iB} being again the values at a certain position in fingerprint A and fingerprint B, respectively. *sim* thus is between ∞ and 0 (identity).

5.2.3 Cosine Coefficient

The similarity value is determined with

$$sim = \frac{\sum_{i=0}^{n} x_{iA} x_{iB}}{\sqrt{\sum_{i=0}^{n} x_{iA}^2 \sum_{i=0}^{n} x_{iB}^2}} \tag{12}$$

with x_{iA} and x_{iB} being the values at a certain position in fingerprint A and fingerprint B, respectively. For continuous variables, *sim* is between 0 and 1 (identity; theoretically also -1 would be possible).

5.3 Clustering

DAIM offers two possibilities to cluster molecules: Jarvis-Patrick clustering, which is appropriate for smaller sets and can be tuned quite effectively, and leader clustering, which is suitable for larger sets, but is not invariant to the sequence of the molecules. The result of the procedure is written to the file `daim.clus` (cf. 9.8.3). Please note that you have two options with which you can call the clustering procedures: `--`cluster and `--`cluster-from-file (cf. 8.2).

5.3.1 The Jarvis-Patrick Algorithm

The *Jarvis-Patrick* algorithm [15], is a one-pass, nonhierarchical method yielding non-overlapping clusters. It has reportedly shown good results when clustering molecules [16]. For DAIM, the modifications suggested in [17] were implemented.

Jarvis-Patrick clustering functions in two stages. In the first stage, the similarity of every molecule (i.e. its fingerprint) to every other molecule is calculated. This results in a $N \times N$ matrix, which contains the values of the Tanimoto coefficient (cf. 5.2.1). In parallel, the *nearest-neighbour list* for every molecule is filled: if the mutual similarity of two molecules is higher than a threshold (which can be set with the parameter *Jarp.Threshold*, cf. 7.3.1), they are inserted into each others nearest-neighbour list. This yields a list of different length for every molecule.

The second stage of clustering uses these nearest-neighbour lists to assign molecules to clusters. At first, every compound forms an individual cluster. Then, two molecules A and B cluster together if they have at least K_{min} of their nearest neighbours in common. In that case, B, and all members of the cluster B is currently in, are assigned to the cluster of A. In the DAIM implementation, K_{min} is calculated individually for every pair of molecules as a percentage of the length of the smaller list (parameter *Jarp.Percentage*, cf. 7.3.1).

5.3.2 The Leader Algorithm

A very reduced and memory-efficient version of lead clustering has been implemented in DAIM in order to be able to cluster huge databases of several million molecules. The first molecule in the list of molecules to be clustered is assigned to the first cluster and treated as its cluster center. The next molecule is compared to the first one and, if more similar to the first one than *Lead.Threshold*, inserted into the first cluster. Otherwise, it opens a new cluster and becomes its cluster center. This continues for all other molecules, each molecule being compared to the cluster centres of already known clusters and either being added to a cluster (if similar enough) or opening up a new one. In that way, non-overlapping clusters with a maximal radius of *Lead.Threshold* are generated. The disadvantage is that for very homogenous distributions of molecules (in fingerprint space), the clustering depends on the order in which the molecules are tested.

5.4 Filtering

The filter was implemented as described in [18], with minor modifications for the '!' cases and the additional condition that looks for multiple instances of one fragment. There are four different filter modes that can be applied to each molecule (each with its own characteristic) which can be combined to form a powerful search tool.

- *Filtering for a fragment:* Ligands will be selected according to whether or not they contain a certain fragment. The fingerprint given as argument has to have a match among the fragments of the ligand with a similarity value greater than or equal to the respective cutoff. This comparison always uses the Tanimoto coefficient (5.2.1). It will work even if a fragment you are filtering for will be deselected during decomposition for other reasons (cf. 6.2).

- *Filtering for an arbitrary substructure:* This mode was implemented according to [19], adapting the VFLib Graph Matching Library, version 2.0 [20]. The algorithm identifies a graph-subgraph isomorphism, if existing. The input is a `.mol2` file of the desired substructure. A match is always exact, i.e., every atom and bond of the query (CHARMm atom types and bond types + ring status) must have a corresponding atom or bond in the ligand. There's only one exception: if the query substructure is linear, the ring status doesn't have to match. Hence, DAIM will also accept molecules in which the (linear) query is part of a ring.

  To give additional flexibility, certain wildcards are allowed as atom types in the query (case sensitive): 'Y' will match an**y** heavy atom, 'A' will match an **a**cceptor and 'D' will match a **d**onor.

- *Similarity to a known ligand:* Although this mode also works by specifying a fingerprint, it is used in a slightly different manner by DAIM. It is invoked if the identifier preceding a fingerprint is enclosed in square brackets. The fingerprint of the entire current ligand is then compared to

the fingerprint in the filterfile using the measure chosen by the user, applying the corresponding similarity cutoff (7.3.1) to judge the quality of the match.

- *Selection based on "cutoff rules":* DAIM can apply cutoffs to various molecular descriptors. At present, 2 rule sets are implemented: the "Lipinski rules" and the cutoffs described by Veber *et al.* [5].

  The "Rules of 5" were found by Lipinski and coworkers [4] and give an estimation of the oral bioavailability of a molecule. This rule set is activated by the keyword 'lipinski' and uses the values from the article as default. These are: number of H-bond acceptors ≤ 10, number of H-bond donors ≤ 5, molecular mass ≤ 500 and CLogP ≤ 5. The second rule set – keyword 'veber' – was obtained from the publication by Veber *et al.* [5], in which they showed that other molecular properties that have a high correlation with the oral bioavailability of a drug molecule are the number of rotatable bonds (≤ 10) and the total number of H-bond acceptors and donors (≤ 11). Additionally, DAIM lets you apply a filter to the number of rings, for which the default value is (in analogy) ≤ 5. There is also the keyword 'cutoff' available which has no default values and therefore lets you select all thresholds yourself.

  All three setups can be modified in the definition section of the filterfile (cf. 7.6). You can set new thresholds or modify existing ones by specifying the new value with certain key letters. To further increase flexibility, you can use both a $\leq$ and a $\geq$ relation at the same time: if the key letter for a cutoff is lower case, DAIM will assign this value to the lower threshold, if it is uppercase, the upper limit is set. Available keys are 'A/a' for the number of **a**cceptors, 'D/d' for the number of **d**onors, 'W/w' for the **w**eight, 'P/p' for the CLog**P**, 'R/r' for the number of **r**ings, 'B/b' for the rotatable **b**onds and 'H/h' for the total number of **H**-bond acceptors and donors. Default for the lower bounds is always 0. As an example, if you want molecules with at least 2 rings, you have to say `'r 2'`, and if you want the mass to be smaller than 450, the key is `'W 450'`.

As logic operators, && ('and'), || ('or'), ! ('not') and ( ) (parentheses) can be used. As mentioned in 7.6, every operator has to be separated with a blank from the next one. In front of every argument a number n can be given, DAIM will then filter for molecules that contain at least n instances of this fragment. If you give a number in front of an identifier for a whole molecule or a substructure, it will be ignored, however.

It is in principle possible to use any combination of the operators and you do not have to use every argument that you listed in the definition section. The !-operator and a number, however, cannot be used in front of parentheses. Thus, instead of

`A || 2 ( D && E )` you have to write `A || ( 2 D && 2 E )`.

5.5 Chirality

The R/S label of a chiral atom is determined employing the algorithms of Labute [21] and Cieplak and Wisniewski [22].

Labute's method iteratively assigns priorities to all atoms of a molecule. In the DAIM implementation, atoms are sorted according to their order number (since information about isotopes is not contained in `.mol2` files). After this first stage, atoms are reordered according to their neighbours and new order numbers are assigned. This is done until every atom either has a unique order or until the sorted list does not change anymore.

Cieplak and Wisniewski connected the "sign of space" to the sense of rotation around a chiral atom. The sign of space is defined as the determinant of a matrix of the x, y, z coordinates of the 4 neighbours:

$$SIGN \begin{vmatrix} X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 \\ X_3 & Y_3 & Z_3 & 1 \\ X_4 & Y_4 & Z_4 & 1 \end{vmatrix} = Z_1 \begin{vmatrix} X_2 & Y_2 & 1 \\ X_3 & Y_3 & 1 \\ X_4 & Y_4 & 1 \end{vmatrix} - Z_2 \begin{vmatrix} X_1 & Y_1 & 1 \\ X_3 & Y_3 & 1 \\ X_4 & Y_4 & 1 \end{vmatrix} + Z_3 \begin{vmatrix} X_1 & Y_1 & 1 \\ X_2 & Y_2 & 1 \\ X_4 & Y_4 & 1 \end{vmatrix} - Z_4 \begin{vmatrix} X_1 & Y_1 & 1 \\ X_2 & Y_2 & 1 \\ X_3 & Y_3 & 1 \end{vmatrix} \quad (13)$$

If $SIGN|V| < 0$, then the atom is marked as R, otherwise, if $SIGN|V| > 0$, it is S. About the limitations of this method, see 11.

5.6 Binding Site

5.6.1 Definition

The binding site is determined by placing a zone (not sphere !) around the original conformation and position of the bound ligand. This pose has to be passed to DAIM with `--`native (cf. 8.2). The zone's size can be set with the *BS.Zone_Radius* parameter (cf. 7.3). DAIM determines the residues for which the percentage of atoms that lie inside the zone equals or surpasses the *BS.Ratio_Inside* (cf. 7.3). With this method it is omitted that large rests of some residues stick out away from the binding site. This leads to a more precise definition of the binding site. Alternatively, you can provide the definition in a separate file (`--`bsfile option, cf. 8.2). It is recommended to take a look at the binding site, which is possible by using the option `--`bsdef (cf. 8.2 and 9.5).

5.6.2 Flexibility

To support the sidechain flexibility which will be allowed in FFLD, DAIM is able to identify the rotatable bonds of the binding site residues. It applies the same criteria as usual (double, triple, aromatic and terminal bonds as well as bonds in rings and in the backbone are considered as being unbreakable) and prints this information into the `@<DAIM>BS` section of the `.info`-file. This mode is activated by using `-R` or the `--`flexreceptor option (cf. 8.2).

6 Decomposition

6.1 Algorithm

The decomposition proceeds in 4 phases: ring identification, initial fragment definition, functional group merging and completion of the valences.

(i) *Rings* are identified by successively enumerating all neighbours (i.e., directly covalently bound atoms) of every atom, similar to a breadth-first search. A neighbour with an already assigned number indicates a ring closure and the corresponding ring size is the sum of the two numbers.

(ii) A *fragment* is defined as a set of atoms connected by unbreakable bonds. The basic definition of unbreakable bonds includes terminal, double, triple, aromatic bonds and bonds in rings. Non-rotatable and unbreakable bonds are distinguished in DAIM; a non-rotatable bond is always unbreakable, whereas the reverse is not true (e.g., a double bond is non-rotatable and unbreakable whereas an amide bond is unbreakable, but can assume more than one conformation). In the default parameter file, an extended definition of unbreakable bonds is used (cf. 7.3.3) since with the basic definition mentioned above single bonds of groups which form chemical entities would be cut (e.g., in a sulfonamide group, the bond between sulfur and nitrogen is formally a single bond and would thus be cut). This extended list includes amide, phosphate group and sulfonamide bonds as well as the single bonds in conjugated systems and the single bond connecting an amidine group. If the parameter *Frag.OverrideDefaultBondDefs* is set, the basic definition is ignored and only the bonds of the extended list are used.

(iii) To form chemically more relevant fragments and avoid the generation of many small groups, fragments corresponding to certain *functional groups* (predefined in `@<DAIM>TABLE`, cf. 7.3.2) are merged with the next fragment they are connected to. This process is unambiguous, since all functional groups defined so far have only one connection point and thus only one neighbouring fragment.

(iv) In the final step, missing atom neighbours are added. An atom will lack a neighbour atom where the bond connecting them has been cut. These missing neighbours are replaced by hydrogen atoms to reconstitute the correct valence for every atom. A methyl group is used to fill valences where a hydrogen atom would result in an unwanted additional hydrogen bond direction (e.g., a hydrogen replacing a carbon atom bound to an sp^3 nitrogen). This is the case for nitrogen, sulfur of the CHARMm atom types `SO1`, `SO2`, `SO3`, and `SK` (or Sybyl atom types `S.3`, `S.o`, `S.o2`, and `S.2`), phosphorus of the CHARMm atom types `PO3` and `PO4` (or Sybyl atom type `P.3`)and oxygen. If you want to switch off the methyl addition, you can do so by setting the parameter *Frag.Smart_Completion* (cf. 7.3.1) to 0. In that case, DAIM will always use hydrogens to complete valences.

These additions mean that new atoms (which were not present in the original molecule) may appear. In the case of hydrogens this is not too problematic, since they are not considered in the fragment definition. If a carbon atom is added, however, it will be part of the fragment definition.

6.2 Fragment Choice

The fragments appearing in the SEED input file are chosen in a 4 (5)-step selection process. DAIM attempts to find a balance between fragments that have many interesting features and those that in all likelihood won't make good interactions.

1. Should the user have specified a list of fragments which are to be excluded (with `--exclude`, cf. 8.2), these are compared to the found fragments and the matching ones are deselected.

2. DAIM computes the "chemical richness" ρ_χ for every fragment that was not deselected so far. This score is the sum over all values in the fingerprint, i.e, $\sum_{i=1}^{17} fp_i$, but neglecting methyl groups or hydrogens added by DAIM. The assumption lying behind this score is that the fingerprints include fields that count both size features and functional groups. The size of a fragment determines in which pockets of a binding site it can fit. The functional groups are likely to form directional interactions and thus determine the orientation. Selecting the fragments with the largest sum over all entries means selecting bigger, "chemical-feature-rich" fragments. All fragments that have a score lower than the parameter *Frag.Min_Size* (cf. 7.3.1) are discarded for reasons of computational efficiency. Should the application of this rule yield fewer fragments than requested by the parameter *Frag.Threshold*, the user will be informed about this in the log file.

3. In the next step, fragments with several substituents are eliminated. These "central" fragments, if highly substituted, will not form significant interactions with the protein for steric reasons. For a cyclic fragment, the number of substituents (n_{subst}) and the number of rings (n_{rings}) are counted and the cyclic fragment is considered "central" if

$$n_{subst} \geq Frag.Substituent\_Factor\_Ring \cdot (n_{heavy\ atoms\ in\ ring} - n_{rings}) \tag{14}$$

   Using a k_r value of $1/1.75$, a disubstituted benzene is not deselected whereas a trisubstituted one is considered "central" and therefore not used as anchor. An acyclic fragment is deselected if

$$n_{subst} \geq Frag.Substituent\_Factor\_Linear \cdot n_{heavy\ atoms} \tag{15}$$

   The default k_l value of 0.5 permits the selection of terminal amide groups (i.e., connected to one other fragment), but rejects amide groups originating from within the chain (i.e., connected to two fragments).

   $Frag.Substituent\_Factor\_Ring$ and $Frag.Substituent\_Factor\_Linear$ are user-definable parameters (cf. 7.3.1). This criterion is only applied if more than *Frag.Threshold* fragments remain. Otherwise, this rule is skipped, which will be mentioned in the log file. These two parameters are also a good way to force DAIM to select "as many rings as possible" or "as many linear fragments as possible": just set $Frag.Substituent\_Factor\_Linear$ or $Frag.Substituent\_Factor\_Ring$ to 0, respectively. The right hand side of equation 15 will *be* 0 and the right hand side of equation 14 will be *set* to 0, thus that kind of fragments will be deselected. If there would be too few fragments of a kind in a molecule, this rule is skipped anyway, thus you do not risk to loose any molecules.

4. If *Frag.Threshold* in `daim.param` is different from 0, DAIM can suggest two sets of fragments. By default, the "most interesting set" (mis) is defined, which contains the top *Frag.Threshold* fragments (ranked according to ρ_χ). If `--mds-also` or `--mds-only` is set, the "maximum diversity set" (mds) is defined. It contains those fragments for which the sum of all pairwise Euclidian distances is largest.

5. In case there is more than one instance of a certain fragment type, DAIM can try to select the fragment that is furthest away from the center of the molecule. This rule is not switched on by default, but one can do so by setting *Frag.RefineSets* to 1 (cf. 7.3.1).

In the `receptorname_seed.inp`-file every fragment *type* will be mentioned only once, though, even if there is more than one fragment of a kind in the molecule (e. g. a ligand with two or more benzenes).

7 The Input Files

DAIM needs a minimum of four input files: the molecule (or a list of filenames of molecules) that shall be decomposed into fragments and the parameterfile set (consisting of `daim.param`, `daim.prop` and `daim.weight`). If available, a database with which the fragments shall be compared will be read. In case there is no such file or it is empty, the database will be generated on the fly. If a SEED input file is to be generated, DAIM also needs a `receptor.mol2` file, the name of which can be specified with the -r-option (cf. 8.2). Optional are furthermore a file containing the specifications for a filter (cf. 7.6) and a file containing an alternative binding site.

7.1 The Molecule

The input file must be in `.mol2` format. DAIM will use the filename (without the `.mol2` ending) for naming all output files. To omit files being overwritten, this should be a unique name. For the determination of rotatable bonds, the *atom name* should be of the form `ELEMENTNAME#`, where `#` is a number. In this way, every atom should receive a unique name. The *atom types* are required to be in either CHARMm or Sybyl format.

Please notice that DAIM does **not** automatically check whether all these rules are being followed or not (since they mostly just ensure compatibility with SEED/FFLD and do not influence DAIM behaviour). You can perform such a check with the `--`check option, however. Should any errors have been found, files can be cleaned with `--`clean (cf. 8.2 for both options).

The DAIM approach is independent of the 3D-structure, but DAIM cannot cope with multiple conformations that are stored in one file.

7.2 The Database

By default, the filename for the database is set to `daim.db` ("database of fingerprints"), but any other name can be given with the `--`dbfile option in the command line. It will be expected to be located in either `DB_DIR` (cf. 7.3.4) or `~/.daim`.
The database has to contain in each line:

- the *filename* of the fragment with which the found fragments shall be compared, followed by
- the *fingerprint* (cf. 5.1), which is supposed to consist of 17 numbers (cf. 10).

7.3 The Parameterfile

This file – called `daim.param` by default – is the main place to change the behaviour of DAIM and set variables which do not change with every run. You are strongly encouraged to take a look at the file that you obtained with the distribution and make sure that all things set there make sense for your project. DAIM will first look for this file in `./`, afterwards in `~/.daim`. Since DAIM 5.0, parameter files have a version number to ensure compatibility of the format with the executable you use. If DAIM complains, just get the latest version of the parameter file.

Reading of the parameterfile relies on the section tags and their appropriate order. No blank spaces are allowed within a field and each field must contain an entry. Entire lines can be commented with '#' (no blank lines allowed, however) and each section has to end in one or more '='. At present, the parameterfile consists of four sections (cf. Table 2):

7.3.1 `@<DAIM>PARAM`

In the first part it is possible to adjust the behaviour of DAIM with the following keywords (case insensitive). In brackets, required input values and defaults (bold) are given.

- *Back_Check* (0/**1**):

  This will cause DAIM to add new fragments (i. e. those that do not have a similarity higher than the *Cutoff_for_Similarity* to any molecule in the database) to the database in the memory of the

program (the information will not be written to any file). Thus, similarity searches will not only be performed in a given library, but also amongst already processed fragments. This ensures that every fragment written to a `.mol2` file in `./Inputs/` is unique.

- *BS.Protein_Level* (int, **3**):

  The value of this parameter is used to insert the appropriate number of levels (i.e., '/' characters) in the `load.wnp` macro (cf. 9.5).

- *BS.Ratio_Inside* (double, **0.5**):

  For the definition of the binding site (cf. 5.6.1) one can additionally specify the percentage of atoms – in each residue – that have to lie within the zone here.

- *BS.Zone_Radius* (double, **5**):

  Setting this variable sets the size of the zone around the ligand which defines the binding site (cf. 5.6.1). This value is given in Å.

- *Cutoff_for_Similarity* (double):

  With this variable the minimum (maximum) value of *sim* (cf. 5.2), meaning that two molecules are considered matching, can be given. Default values are **1** for the Tanimoto and Cosine coefficients and **0** for Euclidian metric.

- *FFLD.Number_GeomCent* (int, **20**):

  The number of anchors FFLD will use to orient the ligand in the binding site.

- *FFLD.OldInfoFile* (**0**/1):

  Set to 1, this parameter ensures compatibility with the older versions of FFLD for which you need an input file (and SFI).

- *Frag.Central_Check* (0/**1**/2):

  This tells DAIM whether it should apply rule 3 (the "central check") of the fragment choice procedure (cf. 6.2) never (0), as defined in 6.2 (1) or always (2).

- *Frag.Min_Size* (int, **10**):

  A criterion for fragment choice (cf. 6.2, rule 2). DAIM will consider all fragments whose fingerprint cross sum is larger than this value further and apply the other criteria.

- *Frag.OverrideDefaultBondDefs* (**0**/1):

  Ignore the default unbreakable bond definitions (double, triple, aromatic, terminal bonds and bonds in rings; cf. 6.1) and use only the definitions provided by the user in the `@<DAIM>NONROT` section of the parameter file (cf. 7.3.3).

- *Frag.Refine_Sets* (**0**/1):

  Try to select the fragments furthest away from the center of the molecule (cf. 6.2, rule 5).

- *Frag.Rename_Atoms* (0/**1**):

  If set, DAIM will assign a new and sequentially numbered name to every atom. Else, atoms of a fragment retain the names they had in the parent molecule. The only exception are atoms added by DAIM which will still receive a sequentially numbered name. DAIM does not check for uniqueness of the atom names, however!

- *Frag.ResetPartialCharges* (0/**1**):

  Choose whether DAIM will replace all partial charges in output `.mol2` files with 99.9999 (value 1) or not (value 0; cf. 11).

- *Frag.Smart_Completion* (0/**1**):

  DAIM will complete valences for all atoms in a fragment (since some of them might lack a binding partner where a bond has been cut). Normally, this is tried to be done in such a way that no additional hydrogen bond donors are generated (cf. 6.1). By setting this variable to 0, you can force DAIM to use only hydrogens for completion.

- *Frag.Substituent_Factor_Linear* (double, **0.5**):

  This factor is used during the selection process for the fragments and affects only fragments that *do not* contain a ring. The number of heavy atoms in a fragment is multiplied with this number and if the number of substituents is greater than the product, the fragment will not be chosen. The default value of 0.5 will *deselect* amide groups in the backbone of a peptide. By increasing this value their choice can be allowed, though. If you set it to 0, all linear fragments will be deselected, provided that enough fragments remain to satisfy *Frag.Threshold* (cf. 6.2, rule 3).

- *Frag.Substituent_Factor_Ring* (double, **2**):

  The counterpart of the above factor for rings. The number of rings is subtracted from the number of heavy atoms in the ring and divided by this number. The default 2 will deselect 3-substituted benzenes, but allow 2-substituted ones. If you set it to a very high number, all fragments with rings will be deselected, provided that enough fragments remain to satisfy *Frag.Threshold* (cf. 6.2, rule 3).

- *Frag.Threshold* (int, **3**):

  DAIM will not define more fragments per molecule than given with that number. If you set it to 0, *all* fragments will be analysed and listed in the output file.

- *Input.Force_Field* (int, 0/**1**/2):

  Atom types of input (and output) molecules (0 CHARMm, 1 TRIPOS, 2 MATCH)

- *Jarp.Percentage* (double, **0.5**):

  Used during Jarvis-Patrick clustering (cf. 5.3.1) to compute the number of nearest neighbours that two structures have to have in common to cluster together.

- *Jarp.Threshold* (double, **0.95**):

  During Jarvis-Patrick clustering (cf. 5.3.1), this is the minimum value of the Tanimoto coefficient such that two molecules are considered as being close.

- *Lead.Threshold* (double, **0.95**):

  During leader clustering (cf. 5.3.2), this is the minimum value of the Tanimoto coefficient such that two molecules are considered as being similar.

- *Qsar.Min_Occur* (double, **0.3**):

  This value specifies the minimum percentage of molecules that contain a certain fragment such that it will be considered as a suggestion in the filter on its own. All the other fragments (with a ratio *smaller* than *Qsar.Min_Occur*) will be subjected to the clustering procedure first (cf. also the option `--`makefilter in 8.2).

- *Quit_on_Hit* (0/**1**):

  Switching on this parameter causes DAIM to advance to the processing of the next fragment if it finds a perfect match in the database. This feature eliminates problems with duplicate entries in `db.fp` and is forced to 1 if the `--`qsar option is set.

The following parameters are used solely to generate the SEED input file, they do not change the behaviour of DAIM. Please also refer to the SEED documentation and [1] for details on what these parameters mean.

- *SEED.Polar_Vectors_Kept* (double, **0.8**):

  Specifies the percentage of polar vectors kept from all possible vectors.

- *SEED.Apolar_Vectors_Kept* (double, **0.8**):

  Sets the size of the ensemble of apolar vectors that are kept for further calculations.

- *SEED.Number_of_Clustermembers* (int, **5**):

  The number of cluster members that will be kept in each cluster.

- *SEED.Sphere_Radius* (double, **10.0**):

  The radius of the cutoff sphere that can be specified. The center of the sphere is determined by DAIM and is the geometrical center of the heavy atoms of the decomposed molecule.

- *SEED.Energy_Cutoff* (double, **5**):

  The cutoff in docking energy SEED uses for the seeded fragments.

- *SEED.2nd_Clustering_Cutoff* (double, **5**):

  The cutoff for the 2^{nd} clustering of the fragments.

- *SEED.Dielectric_Constant* (double, **2.0**):

  Giving this value means that you set the dielectric value of the solute in the SEED input file. Default value: 2.0.

7.3.2 `@<DAIM>TABLE`

In this section, a list of functional groups can be given. They are specified via a count of the different atom types. The names for the groups are just to help the user to distinguish them, DAIM doesn't need them (although it expects an entry there). The definitions are used *before* all the bonds are assigned to the fragments, so functional groups can only consist of atoms connected via unbreakable bonds (for a definition of unbreakable bonds cf. 6.1). A fragment is identified by comparing the corresponding fields of its fingerprint (cf. 5.1) to the values in the list using the Tanimoto coefficient (cf. 5.2.1).

Functional groups are automatically attached to the neighbouring fragment. Depending on the last column (Rot), the new bond (which connects the functional group to the next fragment) is marked as rotatable (1) or not (0). DAIM reiterates through the list of fragments if it encounters an `-OH` in order to be able to spot `-COOH`-groups.

7.3.3 `@<DAIM>NONROT`

DAIM distinguishes between bonds that are rigid and bonds that cannot be cut. The majority of rigid bonds will also be unbreakable, exceptions are possible, however. DAIM automatically treats double, triple, aromatic, terminal bonds and bonds in rings as rigid and does not cut them. With the list given in this part you can add definitions of bonds that ought to be kept rigid and/or should not be cut. Maybe the most important entry here is the one for amide bonds, which was hard-coded until DAIM 4.1, but was moved to the parameter file due to public wish. Please note that bonds that cannot be cut (usually the ones which are considered to be rigid) are used to define the fragments, therefore the bond types given here have an influence on the decomposition (cf also 6.1)!

In the definition, after a name, the bond type in `.mol2` format has to be given. In the fields "AtomO1", "AtomM1", "AtomM2" and "AtomO2" (the **o**uter and **m**iddle atoms of the dihedral defining the bond, Fig. 2) appropriate atom types can be listed. If more than one atom type is given, each has to be separated with '/' from the next. '*' can be used as a wildcard. Both the left and the right atom of a bond are compared to both lists (AtomM1, AtomM2). But only if the left and right atom match in different lists, the bond is considered further. In the next step, DAIM investigates if any neighbour of AtomM1 is of type AtomO1 (and vice versa for the neighbours of AtomM2). "Rot" specifies whether this bond should be considered as rotatable or not (1/0, resp.) and "Cut" tells DAIM if it can cut this bond (0/1).

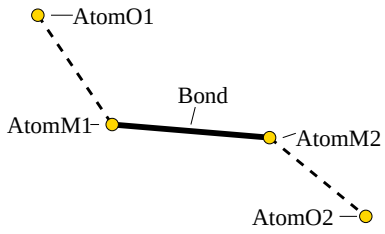

Figure 2: Definition of a dihedral.

7.3.4 @<DAIM>CONFIG

To allow the user to structure the in- and output in a clear way and facilitate the screening of a large library, several target directories for DAIM in- and output can be listed here. The syntax is a (case insensitive) keyword followed by the name of the directory, both absolute and relative paths are possible. Each output directory will be automatically generated, DAIM is not able to generate a whole tree of directories, though. The default for all directories is '`./`', except for `SEED_MOL2_DIR`, for which it is '`./inputs`'. Available keywords are:

`BASE_DIR` (directory relative to which all other directories will be created)

`DB_DIR` (directory from/to which the database will be read/written)

`HBO_DIR` (directory to which the `.hbo` files will be written)

`HTML_DIR` (directory to which the output `.html` files will be written)

`INFO_DIR` (directory to which the `.info` files will be written)

`LIG_DIR` (directory from which the `molecule.mol2` file will be read)

`NAT_LIG_DIR` (directory from which the native ligand(s) will be read)

`PROT_DIR` (directory from which the `receptor.mol2` file will be read)

`SEED_INP_DIR` (directory to which the `seed.inp` file will be written)

`SEED_MOL2_DIR` (directory to which the `.mol2` files of the fragments will be written)

7.4 The Propertyfile

`daim.prop` is located in `~/.daim`, too, and is a list of some of the properties of the different atom types. It is based on the CHARMm MSI parameter set and uses its atom types as well as Sybyl atom types. For each atom type, it lists the average mass, the order number, the electronegativity according to Pauling and the covalent radius. DAIM uses this list to compute the molecular weight of a compound, to distribute initial order numbers when investigating the chirality of atoms (cf. 5.5) and to compute the Wiener 4 Index (cf. 5.1.3).

7.5 The Weightfile

Another file that resides in `~/.daim`. It contains two columns for weights for each of the fields of a fingerprint. The weights in the first column are used when DAIM computes the "chemical richness" ρ_χ for the fragments and molecules (cf. 6.2). Alteration of the weights means that the selection of the *Frag.Threshold* most interesting fragments might change. These weights are useful if you want to emphasise the importance of certain features in the fragment selection procedure. By default, all weights are set to 1.

The weights in column two affect the calculation of molecular similarity. They are used as normalization factors (i.e., each field is multiplied by the value in column two) in order to limit the contributions of certain fields. The default weights have been taken from reference 23.

7.6 The Filterfile

This file has two sections (Table 3): in the first section – the "definition section" – fingerprints (cf. 5.1) for the fragments or molecules you want to use as arguments for the filter have to be listed. Each fingerprint has to be followed by the cutoff value for the similarity (defined by the Tanimoto coefficient), thus allowing different strictness in identifying the various fragments. The second section – the "command section" – (separated from the definition section with '#') consists of a line which tells DAIM in which way you want to combine the fingerprints to form the filter.

It is possible to use any kind of string as identifier for the fingerprints. All identifiers are case insensitive. To allow DAIM to distinguish between the fingerprint of a fragment and the fingerprint of

a whole ligand, the latter must be enclosed in square brackets. The filename of a substructure needs not be mentioned in the definition section, it is enough to include it in the command section. It has to contain the string '.mol2' to be recognised, though. The command line can consist of any number of logic operators, but each operator must be separated by *one* blank from the following. For more information on the different filter modes, please refer to chapter 5.4.

8 How to Run DAIM

8.1 Command Line

DAIM can be run with

`daim_x -options -i molecule.mol2 [--html]` (single molecule mode) or
`daim_x -options -c# --list mol.list [--html]` (multi molecule mode, `#`=number of molecules)

If you run DAIM without any options, the program will normally decompose the specified molecule and compare it to the default database using the continuous Tanimoto coefficient. Only unmatched fragments are written to `.mol2` files then.

8.2 Options

The options available are:

- *append* -a, --append

  The fingerprint of the (whole) molecule given in the command line is computed and added to the database file. In this mode, the molecule is **not decomposed** !

- *Bertz index* --bertz

  The Bertz index (cf. 5.1.4 and [11]) will be computed and written to stdout. Otherwise, nothing is computed.

- *binding site definition* --bsdef

  The definition of the binding site has a major influence on the results of the docking calculation. It is thus advisable to evaluate this definition in detail (cf. 5.6.1). This option will take only the protein and the native ligand(s) as input, but no molecule to decompose and will generate three output files, `load.wnp`, `load.vmd` and `bs.def` (cf. 9.5).

- *binding site* --bsfile [`site`]

  If you are not satisfied with the calculated binding site or if there is no co-crystallisation structure available, you can pass a list to DAIM here. This list has to contain the sequence numbers of the residues and needs **not** be preceded by the *number* of residues. Please note that this option also needs -r (or-R) and -s (cf. there and also 9.6). When passing the definition of a flexible binding site, the residue number must be followed by the number of rotatable bonds on the same line and by the bond definitions in the following ones.

- *check file* --check

  A thorough check of the molecule(s) given in the command line is performed. DAIM will check whether the file is compatible with FFLD, fulfils all the requirements listed in 7.1, contains more than one molecule or is strange in any other way. The result of this check is printed into the log file (cf. 9.2.1), otherwise no output is generated. If there are problems which can be solved by DAIM (see next option), this will be stated.

- *clean file* --clean

  If DAIM diagnosed some solvable problems, they can be eliminated in the respective files with this option. Problems that can be solved are mainly naming and counting problems, but FFLD tends

to be quite intolerant towards those. The output is a file `ligand_c.mol2` (i.e. with '_c' appended to the original filename).

- *cluster molecules* -j [mode], --cluster [mode]

  A list of molecules passed to DAIM is clustered according to their fingerprints using the specified algorithm (mode): `j` → Jarvis-Patrick algorithm (cf. 5.3.1) or `l` → leader algorithm (cf. 5.3.2). A list of the clusters and the representatives is written to `daim.clus` (cf. 9.8.3).

- *cluster a list of fingerprints* --cluster-from-file

  Same as the previous option, but takes a file containing the precalculated fingerprints as input. This file can be prepared using -d and abolishes the need to reread the molecules every time.

- *print information about connection points* --connection-info

  When run in this mode, DAIM prints information about connection points and newly added atoms to the `.mol2` files of the fragments. When a bond is cut, the letter 'x' is appended to the name of the atom remaining in the fragment (i.e. the atom which now lacks a neighbour). DAIM will reconstruct this missing neighbour (cf. 6.1) and the letter 'a' will be appended to the atom name (if a $-CH_3$ is added, every atom will be marked with an 'a').

- *count of input molecules* -c#, --count#

  Used for the decomposition of more than one ligand. In the SEED input file (if wanted), every fragment type resulting from the decomposition is listed only once. The `ligand_m[i,d]s.info`-files do contain the correct specifications, though. The list of ligands has to be redirected to the executable, each filename in a new line. '#' has to be replaced with the number of ligands.

- *molecular property file generation* --dat

  A useful option to generate *only* `daim_all.dat` (cf. 9.8.1) – except for the log file no other directories or files are generated.

- *database generation* -d#, --dbgen#

  This mode is to be used for the generation of a database of fingerprints, whose filename can be given. '#' stands for the number of files in the database. The list of the filenames has to be redirected to the program (each filename in a new line). With this option the only output is the database and the molecules are **not decomposed** !

- *database of fingerprints* --dbfile [database]

  If you want to use your own database of fingerprints instead of `db.fp`, you can enter the name here.

- *exclude* --exclude [list]

  This option is another possibility to influence the choice of the fragments (cf. 6.2, rule 1): in `list` you can list fingerprints (format: string + 17 floats) which DAIM will deselect when it encounters them during the decomposition. Useful to modify the fragment selection if SEED couldn't place a certain fragment.

- *exhaustive set definition* --exhaustive-sets

  If you are neither satisfied with the fragment selection in the mis nor the one in the mds (cf. 6.2, rule 4 and below), this option will provide you with *all* possible combinations of fragments. No rules from 6.2 will be applied. Bear in mind that this will yield a potentially very large number: there are 120 possible triplets in a molecule with 10 fragments.

- *force field* --fftype#

  Specify the atom types of input and output molecular files.

- *filter* -f, --filter [filter]

  Invoke the filter defined in [filter] (cf. 7.6 and 5.4).

- *filter only* --filter-only [`filter`]

  This option is intended for the rapid screening of libraries. All steps in DAIM to define and process fragments will be skipped, consequently it is only possible to filter for entire molecules, exact substructures and cutoffs. In all other cases, the option will be changed to --filter.

- *H-bond information* --hbo

  If switched on, DAIM will generate a file `ligand.hbo` containing the names of acceptor and donor atoms (cf. 9.4).

- *help* -h, --help

  Print the available options.

- *html files* --html [`html`]

  Tell DAIM that you want output in HTML-format and specify the output file name.

- *molecular file* -i, --mol [`molfile`]

  Give the name (path) of the a molecular file.

- *list file* --list [`listfile`]

  Give the name of the file that lists names (paths) of molecular files.

- *log file* --logfile [`logfile`]

  Give the name of the file you want DAIM to write the log to (different from the default `daim.log`).

- *filter generation* --makefilter

  This option will try to extract common fragments from a list of molecules. These fragments are then clustered (cf. 5.3.1) and with this information a suggestion for a filter is written to `filter.sug` (cf. also 7.6 and 11).

- *maximum diversity set* --mds-also, --mds-only

  If set, DAIM will suggest an alternative to the default "most interesting set" and write it to a separate `.info` file (cf. 6.2, item 4).

- *metric* -m#, --metric#

  Instead of '#', the metric should be specified: 1 for continuous Tanimoto coefficient, 2 for Euclidian metric, 3 for cosine coefficient. If not specified otherwise, the molecule will be compared to `db.fp`. The output will be sorted according to the metric chosen.

- *compute similarity* --mol1 [file1] --mol2 [file2] *or* --fp1 [file1] --fp2 [file2]

  The program lets you directly compute the similarity between two molecules *or* two fingerprints. The fingerprints have to be stored in a file, not explicitly listed. Please note that the two variants might yield slightly different values (cf. 11).

- *native ligand (list)* --ligand [`ligand(list)`]

  Whenever a binding site definition (cf. 5.6.1) is desired, the molecule(s) around whom the cutoff will be applied have to be included with this option. You can either name one molecule (whose filename has to end in '`.mol2`') or a whole list of molecules (where the filename of the list must *not* contain the string '`.mol2`'). In case more than one molecule is given, the resulting binding site will be the set union of each individual definition.

- *suppress generation of* `daim_all.dat` --nodat

  All other functions work normally, only this file is not generated.

- *parameters* --paramfile [`parameterfile`]

  Together with this option, a different parameterfile than the default `daim.param` can be given.

- *parameters* --propfile [`propfile`]

  Together with this option, a different property file than the default `daim.prop` can be given.

- *parameters* --weightfile [`weightfile`]

  Together with this option, a different weight file than the default `daim.weight` can be given.

- *output in pdb format* --pdb

  Fragments written to files will be generated in `.pdb` instead of `.mol2` format.

- *statistics on the identified fragments ('qsar')* --qsar

  An additional file, called `statistics.html`, is generated. Its content is described in 9.8. This option does not work together with the -w or -a option.

- *receptor* -r, --receptor, -R, --flexreceptor [`receptor.mol2`]

  For the SEED input file (cf. 9.6), a definition of the binding site is necessary. DAIM can compute this information (cf. 5.6.1), using the `.mol2` files of the receptor (given here) and a ligand. Alternatively, you can provide your own definition with --bsfile (cf. there and also -s). The filename of the receptor will furthermore be used to generate the name of the SEED input file. For this reason it should not contain a path.

  If the option is the capital letter, DAIM will try to determine the rotatable bonds for the sidechains of the amino acids in the binding site. For this operation, the same rules as for the rest of the calculation will be used. This information will be written to the `.info`-file and used by FFLD.

- *seed input* -s, --seed

  Here you can set whether or not you want an input file for SEED to be generated (cf. also 9.6). Please note that for a correct input file you need a binding site definition, using either --bsfile and/or --native (cf. also 9.6).

- *seed parameters* --seedpar [`seedpar`]

  Together with this option, a different seed parameter file than the default `seed.par` can be given.

- *show values of all user-changeable variables* --show

  DAIM will print the values of all parameters (cf. 7.3.1) and directories (cf. 7.3.4) to stderr.

- *skip problematic molecules* --skip-unknown-atoms

  Atoms with unknown atom types can cause confusion. Therefore, DAIM will usually exit when encountering problems. With this option set, the program will continue to run, but will skip the molecules containing problematic atoms.

- *project tag* -t, --tag [`tag`]

  Project tag, if given, will be prepended to output files and directory names.

- *unique* -u, --unique

  Test a database for double entries.

- *version* -v, --version

  Prints the version number of DAIM.

- *write* -w#, --write#

  '#' should be replaced by the number of the specific fragment that shall be written to a `.mol2` file or by the keyword 'all' in case you want every fragment. This will work for any fragment regardless of the size. A similarity search will be done. Useful for looking at unknown fragments.

9 The Output Files

9.1 The Output File

You will only generate detailed output if you use the `--html` option. The html output file then contains links to files containing the data for each decomposed ligand, to the final SEED input file and to the file containing the statistics (cf. 9.8). Each ligand's data file then contains

- the values of several descriptors of the molecule
- the fingerprint
- a list of the chiral atoms
- a list of all the bonds and the size of the rings they are part of
- a list of all the found fragments
- a link to the file to which the list of rotatable bonds has been written
- the result of the similarity analysis for the fragments suggested (cf. 6.2).

If the -d or -a option is set, the only output is the generated database. The ligand data files are named using the name given in each ligand's `.mol2` file.

9.2 The Log File

To this file DAIM writes information about the processing of the molecules and which steps it took. I starts by echoing the command line, the values of all parameters and the names of the directories set in `@<DAIM>CONFIG`. At the end there will be a line stating how many of the input molecules have been processed, the rest having been discarded by the filter or skipped because of unknown atoms. Furthermore, it contains the report on a molecule when `--check` was invoked (cf. the following section). It can furthermore be used to get information about the course of the processing of a large database.

However, only limited information on how and why a molecule was decomposed in a specific way can be found. This can be obtained either in the `ligand.html`-file or in this manual.

9.2.1 Information from --check

The option `--check` is useful to get hints on where the problems with a specific molecule might have been. In the log file warning messages will be issued for every problem that was encountered. Furthermore, two lines, each commencing with 'DD' ("DAIM Diagnosis"), will be printed for every molecule:

```
DD: ligand.mol2 PROB 0 SOLV 0
DD: ligand.mol2 AN:0 SU:0 CH:0 NA:0 NB:0 PC:0 CT:0 SN:0 CO:0 HY:0
```

The tags with capital letters can be grepped for. `PROB` stands for the total number of problems, `SOLV` for the number of problems DAIM will be able to solve, respectively. The second line provides an error code for each of the checks DAIM performs. The number which follows will be 0 if this check did *not* detect any error, 1 otherwise. The molecule from the example above thus is error-free.

The codes in detail:

`AN` **A**tom **N**ame: verifies that every atom has a unique name. Solvable.

`SU` **Su**bstructures: the molecule consists of more than one substructure. Solvable. Note that this does not mean that all substructures are connected to each other.

`CH` **CH**ARMm atom types: checks whether all atom types are listed in `daim.prop` (cf. 7.4). Not solvable.

`NA` **N**umber of **A**toms: the number of atoms given in the header is different from the number of atoms really listed. Solvable.

`NB` **N**umber of **B**onds: the number of bonds given in the header is different from the number of bonds really listed. Solvable.

`PC` **P**artial **C**harges: DAIM will complain if all partial charges are 0. Not solvable.

`CT` **C**onnection **T**able: there are bonds which connect non-existing atoms. Not solvable.

`SN` **S**ubstructure **N**ame: for FFLD, this name must not be longer than 4 characters. Solvable.

`CO` **Co**nnectivity of the molecule: DAIM walks over the graph and checks whether it is wholly connected. Not solvable.

`HY` **Hy**drogens: the molecule contains no hydrogen. Not solvable.

With this detailed listing of the errors it is possible to distinguish between molecules which have problems detrimental to the current task and those with mistakes that are of no consequence.

9.3 The `.mol2` or `.pdb` Files

In the case of the -w, -c or -m option, there are furthermore the `.mol2` or `.pdb` files for the fragments that have been written (because this was either specified or they could not be matched) to the directory that was set in the `@<DAIM>CONFIG` section of the parameter file (cf. 7.3.4).

9.4 The `.hbo` Files

For the `--`hbo option, DAIM will produce a file `ligand.hbo` which contains a list of the atom names followed by the donor ('D') and acceptor ('A') properties of each of these atoms. For donors, also the names of the hydrogen atom they are connected to are given.

9.5 Files about the Binding Site Definition

If the option `--`bsdef was given, three files can be found: `daim.wnp`, `daim.vmd` and `daim.bsdef`. `daim.wnp` is a macro for Wit!P which will read the protein and color the binding site residues green. If this is not the case, you have to adjust the parameter *BS.Protein_Level* (cf. 7.3.1), so that it conforms with the structure of the protein. Additionally, it will make the binding site bonds thick and set the pivot point to the center of the protein. `daim.vmd` is a script for VMD, which displays the protein in the same style as Wit!p. Additionally, it will draw the cutoff sphere used by SEED. In `daim.bsdef`, the residue numbers and their names are listed, as well as the parameter values and the filename(s) of the ligand(s) with whom the definition was obtained.

9.6 Files for SEED

For the `-`s option one can find the `receptorname_seed.inp`-file. The latter can directly be used to start a SEED-run. Be aware, however, that depending on the other options this file might lack some information:

- in case you set `-`r and `--`bsfile, there will be no information on the points within the binding site (used for the angle criterion), which would be the coordinates of the heavy atoms of the original ligand(s). Furthermore, the spherical cutoff is omitted, since its center would be the geometrical center of the heavy atoms of this ligand (or these ligands).
- if you specified `-`r and `--`ligand, everything is complete.

SEED can dock fragments in three different modes: **p**olar, **a**polar, and **b**oth. DAIM will suggest 'p' mode if the fragment has a formal charge (even if the net charge is 0); 'b' mode if it contains polar atoms, but has no formal charge; and 'a' otherwise.

9.7 Files for FFLD

These files are called `ligand_mis.info` and/or `ligand_mds.info`, depending on whether they contain the "**m**ost **i**nteresting **s**et" or the "**m**aximum **d**iversity **s**et". They consist of five subsections, `@<DAIM>ATOMS` (containing a link atom and a first forbidden), `@<DAIM>FRAG` (the numbers of the fragment atoms in the original file), `@<DAIM>BS` (the definition of the binding site) `@<DAIM>ROT` (the list of rotatable bonds) and `@<DAIM>ANGLE` (the list of flexible angles; definition pending).

9.8 Statistics

9.8.1 `daim.dat`

This is a text file which contains the number of atoms, bonds, acceptors, donors, fragments and rotatable bonds as well as the molecular weight and the CLogP for the processed fragments.

9.8.2 `daim.stats`

This file (which is generated when the `--`qsar option is set) shows – for each fragment –

- the fingerprint,
- the possible name,
- the number of occurrences,
- the number of molecules it is contained in.

Names are based on the entries in the `daim.db` and are only given if a perfect match was identified during similarity search. In this list only the 'chosen' fragments appear, fragments that were deselected (cf. 6.2) will not be shown.

9.8.3 `daim.clus`

`daim.clus` is generated when the `--`cluster option is used. It lists the clusters and the molecules each of them contains. Every molecule is listed by the order number it had in the input file (commencing with 0). Additionally, DAIM suggests a cluster representative. This is the molecule which is closest to the cluster center (the latter being the average over the fingerprints of all molecules; hence its fingerprint might not correspond to a real molecule).

10 Hardcoded Parameters

At present, there are two parameters that are of interest and can be changed in the files `util.hpp` and `fp_class.hpp` :

- the **number of entries in the fingerprints** (variable name `fp_size`): it is set to ~~17~~23. Changing this value also means that you have to create appropriate member functions to access and set the additional field(s) in the class `FP`.
- the **number of fields for the functional groups** (variable name `fg_size`) is set to 9 (the first 7 fields from each fingerprint plus the number of hydrogens plus the rotatability flag). If you want to add more fields you should also change the number of fields in the fingerprints and make sure that values are assigned to them.

11 Limitations

- partial charges:

  in the fragment `.mol2` files that are generated, if allowed by *Frag.ResetPartialCharges*, DAIM assigns each atom a fake partial charge of 99.99999 to prevent users from directly using them in SEED.

- additional hydrogens and methyls:

  a fragment will always lack an atom at the position where it was connected to any other part of the molecule before. To remedy this, hydrogens are added. Their coordinates are derived from the coordinates of the atom they are replacing by calculating the vector between it and the atom they are connected to. The vector then is normalised to a length of 1.1 Å. If *Frag.ResetPartialCharges* is 0, the partial charge is set to 0.4 if the hydrogen is attached to O, to 0.2 if it is bound to N or S, to 0.13 if it is connected to an aromatic C and to 0.05 in all other cases. These values are close to the corresponding partial charges conforming to the CHARMm22 force field [24]. As CHARMM atom type, `HA` is used.
  When a methyl is added, the C will be of CHARMm type `CT`. The bond length to the next heavy atom will be 1.5 Å. The hydrogens will be of type `HA`.

- CLogP:

  The CLogP is calculated from 14 of the 17 fingerprint fields (NumS, NumP and Wiener 4 Index are not used, cf. 5.1.5). It works quite nicely, but still can never be more than an estimate, since it does not take into account real (3D) structural information and uses only few atom types. This means that a cutoff using the CLogP should never be applied to strict.

- chirality:

  The methods used in Daim are only good for assigning the chirality of single atoms. They can not be applied to get information about the chirality of *molecules*, since no information about symmetries in the molecule are gathered.

- `--`makefilter:

  This option is in its developmental stage. It is not validated and probably buggy. Currently, only fragments are considered. The suggestions for the cutoff (last entry in a line) are based on the results of the clustering. Hence, use at your own risk!

- fingerprints:

  In contrast to their name, the fingerprints are not totally unambiguous: two different molecules might have identical fingerprints (the reverse is not true: molecules with different fingerprints *are* different). However, since the inclusion of the Wiener 4 Index, this affects only few molecules anymore. In a test of the ChemDiv library, only 4 out of 225252 were "false twins". This is less than 0.0018 % and probably acceptable.

- direct computation of the similarity value:

  If you let the similarity value be computed by giving two molecules to the program, Daim will compute the fingerprints itself. Thus, it will use full floating point accuracy for all values. On the contrary, if you specify two fingerprints, Daim can only use them with the given accuracy. This may lead to slightly different values.

```
.DPAF.0708.52.PTNC
# Here you CAN set parameters. All of them have default values.
# WHAT DOES MATTER IS THE ORDER OF THE SUBSECTIONS!
#
@<DAIM>PARAM
Input.Force_Field 1
====================

#Name    NumAt    NumC    NumN    NumO    NumHal   NumS    NumP    NumH    Rot
#
@<DAIM>TABLE
Meth     4        1       0       0       0        0       0       3       0
CHal3    4        1       0       0       3        0       0       0       0
COOH     4        1       0       2       0        0       0       1       1
COO-     3        1       0       2       0        0       0       0       1
SO3      4        0       0       3       0        1       0       0       1
CHO      3        1       0       1       0        0       0       1       1
...
====================

#Name            Bond     AtomO1   AtomM1       AtomM2       AtomO2   Rot     Cut
#
@<DAIM>NONROT
### CHARMm definitions
otheramide       1        *        C            NP/NX        *        0       0
sulfonamide      1        *        NP           SO2          *        1       0
conjugated       1        *        CUA1         CUA2/CUA3    *        0       0
2_triple_bd_C    1        *        CUY1/CUY2    CUY1/CUY2    *        0       0
amide            am       *        *            *            *        0       0
...
====================

@<DAIM>CONFIG
BASE_DIR ./
LIG_DIR ./
HTML_DIR ./html
INFO_DIR ./info
HBO_DIR ./hbo
SEED_MOL2_DIR ./inputs
====================
```

Table 2: A Daim parameterfile

```
A           8 3 1 0 0 1 0 0 2 0 0 1 0 0 5 5 0.208       0.95
C           14 4 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0.523      1.0
Lipinski    W 500.00 A 4 r2
[viracept]  85 32 3 4 0 1 0 12 2 0 2 5 4 0 22 17 8.92   0.93
Benzene     12 6 0 0 0 0 0 6 0 0 0 0 0 0 6 6 0.381      1.0
#
2 A && ( Benzene || C ) && !  [viracept]
```

Table 3: A Daim filterfile

Acknowledgements.
The basic decomposition in DAIM is based on C-code that has been written by A. Widmer at Novartis Pharma AG, Basel. I thank A. Caflisch, F. Dey, D. Huang, M. Cecchini, N. Majeux, P. Alfarano, C. Ehrhardt and S. Jelakovic for useful discussions and tests.

References

[1] MAJEUX, N., SCARSI, M., APOSTOLAKIS, J., EHRHARDT, C., AND CAFLISCH, A., Exhaustive docking of molecular fragments with electrostatic solvation. *Proteins: Structure, Function and Genetics 37* (1999), 88–105.

[2] MAJEUX, N., SCARSI, M., AND CAFLISCH, A., Efficient electrostatic solvation model for protein-docking. *Proteins: Structure, Function and Genetics 42* (2001), 256–268.

[3] BUDIN, N., MAJEUX, N., AND CAFLISCH, A., Fragment-based flexible ligand docking by evolutionary optimization. *Biol. Chem. 382* (2001), 1365–1372.

[4] LIPINSKI, C. A., LOMBARDO, F., DOMINY, B. W., AND FEENEY, P. J., Experimental and computational approaches to estimate solubility and permeability in drug discovery and developmental settings. *Advanced Drug Deliv. Rev. 23* (1997), 3–25.

[5] VEBER, D. F., JOHNSON, S. R., CHENG, H.-Y., SMITH, B. R., WARD, K. W., AND KOPPLE, K. D., Molecular properties that influence the oral bioavailability of drug candidates. *J. Med. Chem. 45* (2002), 2615–2623.

[6] EULER, L., Elementa doctrinae solidorum. *Novi. Comm. Acad. Sci. Imp. Petropol. 4* (1752), 109–140.

[7] YANG, F., WANG, Z.-D., AND HUANG, Y.-P., Modification of the Wiener Index 4. *J. Comput. Chem. 25* (2004), 881–887.

[8] DOWNS, G. M. AND WILLETT, P., Similarity searching in databases of chemical structures. *Reviews in Computational Chemistry* (1996), 1–66.

[9] WIENER, H. J., Structural determination of paraffin boiling points. *J. Am. Chem. Soc. 69* (1947), 17–20.

[10] BERTZ, S. H., The first general index of molecular complexity. *J. Am. Chem. Soc. 103* (1980), 3599–3601.

[11] HENDRICKSON, J. B., HUANG, P., AND TOCZKO, A. G., Molecular complexity: A simplified formula adapted to individual atoms. *J. Chem. Inf. Comput. Sci. 27* (1987), 63–67.

[12] BROTO, P., MOREAU, G., AND VANDYCKE, C., Molecular structures - perception, auto-correlation descriptor and SAR studies - system of atomic contributions for the calculation of the normal-octanol water partition-coefficients. *Eur. J. Med. Chem. 19* (1984), 71–78.

[13] GHOSE, A. K. AND CRIPPEN, G. M., Atomic physicochemical parameters for three-dimensional structure-directed quantitative structure-activity relationships I. partition coefficients as a measure of hydrophobicity. *J. Comput. Chem. 7* (1986), 565–577.

[14] WANG, R., GAO, Y., AND LAI, L., Calculating partition coefficient by atom-additive method. *Perspectives in Drug Discovery and Design 19* (2000), 47–66.

[15] JARVIS, R. A. AND PATRICK, E. A., Clustering using a similarity measure based on shared near neighbors. *IEEE Trans Comp C-22* (1973), 1025–1034.

[16] WILLETT, P., WINTERMAN, V., AND BAWDEN, D., Implementation of non-hierarchical cluster-analysis methods in chemical information-systems - selection of compounds for biological testing and clustering of substructure search output. *J. Chem. Inf. Comput. Sci. 26* (1986), 109–118.

[17] Brown, R. D. and Martin, Y. C., Use of structure-activity data to compare structure-based clustering methods and descriptors for use in compound selection. *J. Chem. Inf. Comput. Sci. 36* (1996), 572–584.

[18] Lippman, S. B. and Lajoie, J., *C++ Primer*, 3^{rd} edition, Addison-Wesley Pub Co (1998).

[19] Cordella, L., Foggia, P., Sansone, C., and Vento, M., An improved algorithm for matching large graphs, in *Proc. of the* 3^{rd} *IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition* (2001), 149–159.

[20] Foggia, P., *The VFLib Graph Matching Library, version 2.0*, http://amalfi.dis.unina.it/graph/db/vflib-2.0/doc/vflib.html.

[21] Labute, P., An efficient algorithm for the determination of topological RS chirality. *J. Chem. Comput. Group* (1996), http://www.chemcomp.com/Journal_of_CCG/Articles/chiral.html.

[22] Cieplak, T. and Wisniewski, J. L., A new effective algorithm for the unambiguous identification of the stereochemical characteristics of compounds during their registration in databases. *Molecules 6* (2001), 915–926.

[23] Dey, F. and Caflisch, A., Fragment-based de novo ligand design by multi-objective evolutionary optimization. *J. Chem. Inf. Model. 48* (2008), 679–690.

[24] Momany, F. A. and Rone, R., Validation of the general purpose QUANTA 3.2/CHARMm force field. *J. Comput. Chem. 13* (1992), 888–900.

A Changelog[2]

Version	Release Date	Features
5.4	01 Aug 2016	--list, --tag, --fftype, --propfile, --weightfile, --seedpar, --mol flags introduced. --native flag changed to --ligand. Support for redirection operators '>', '<' removed. Input.Force_Field parameter introduced. Fingperprint extended to 23 fields.
5.3	07 Sep 2009	Frag.OverrideDefaultBondDefs, Frag.ResetPartialCharges, 'p' mode in SEED input file, removed support for multiple conformations
5.2	06 Jul 2008	support of Sybyl atom types
5.1.4	29 Jan 2008	weights for similarity calculation, two temporary switches for weights: Sim.UseWeights1, Sim.UseWeights2
5.1.3	21 May 2007	bugfix in CollectGraphInformation() for very large molcules, bsdef visualisation as VMD script, SK is acceptor, bugfix for combination of –qsar and –cluster, improved handling of conflicting options, grepable comment tags in html output, size-based handling of the internal database
5.1.2	20 Nov 2006	Frag.Rename_Atoms, bugfix for exhaustive sets
5.1.1	30 Oct 2006	precision bugfix in fingerprint comparison, default for SEED vectors changed to 0.8, bugfix to handle wrong –count number, bugfix for molecule path
5.1	09 Oct 2006	no -CH3 to S except SO2, –exhaustive-sets, –pdb, –show, bugreport, GNU standards compatible distribution, –dat, –nodat, bugfix for broken link in output, leader clustering, daim.weight, bugfix for rotatable bond determination, catch for incorrect count with -c
5.0	20 Jan 2005	.info updated, moleculename from filename, bugfix for -w, crossdock→count, FFLD frag def fix, HTML optional, –connection-info, W4 in fp, –bertz, bugfix for reading filters, nonrot definition as dihedral, new CLogP fit, hash_map for atom properties, –mol1/mol2/fp1/fp2, calculation class, Frag.Substituent_Factor_Linear & _Ring, FFLD compatibility parameters, maximum diversity set, NumRings changed, version numbers for files, donor heavy atoms and hydrogens listed in .hbo
4.3.2	18 Jun 2004	–hbo, daim.prop, –check info restructured, –skip-unknown-atoms, –cluster-from-file, bugfix for CH_3-addition to OS, rot/cut flag for rigid bonds
4.3.1	31 Mar 2004	wildcards in substructure search, –cluster, –unique, keyword 'cutoff', definition of polar fragments changed
4.3	13 Feb 2004	Jarvis-Patrick clustering, –makefilter, new parameter names, fix for db opening, chirality, 'installation' script, man page
4.2.3	09 Feb 2004	fixes for reading filters
4.2.2	27 Nov 2003	Lipinski with upper/lower bound, CLogP calculation from fit to measured LogP
4.2.1	28 Oct 2003	bugfix for reading native ligand, option HTML_DIR
4.2	23 Oct 2003	Smart_Completion, rigid amide user-definable, lines in parameter file commentable, substructure search, number of rings and CLogP in Lipinski, daim_-all.dat, –exclude, bugfix in ring size determination
4.1	Sept 2003	bugfix Lipinski, Lipinski both $\geq$ & $\leq$
4.0.3	27 Aug 2003	hydrogen count bug, central criterion modified
4.0.2	22 Aug 2003	–filter-only
4.0.1	13 Aug 2003	bugfixes in protein reading, frag→molec and protein rotatable bond definition
4.0	11 Aug 2003	config section, –check, –clean, atoms cloned to form fragments, fp for ligands, filter for ligands, cosine coefficient, -w all, strings for parameters, individual sim cutoff in filter
3.5	week 29 2003	determination of protein flexible bonds, longest chain, –bsdef, –native
3.4	27 Mar 2003	Lipinski filter, methyl addition to S
3.3	06 Jan 2003	new QSAR information, Quit_on_Hit
3.2.1	25 Nov 2002	methyl count bug
3.2	18 Nov 2002	-b, definition of fragments, joint fragments, database handling, Back_Check
3.1	04 Nov 2002	log file, improved filter, methyl addition, fix for definition of donor
3.0	17 Oct 2002	implementation of filter, fix for additional donors

[2]For a more detailed description of the changes see the corresponding `ChangeLog` file.

Version	Release Date	Features
2.4	14 Oct 2002	bugfixes in memory management
2.3	week 27 2002	qsar option, redock option, minor adaptations of fingerprints
2.2	week 13 2002	inclusion of parameter file
2.1	week 10 2002	definition of binding site, additional criteria for the selection of fragments
2.0	week 3 2002	processing of multiple molecules, output in HTML-format
1.4	week 48 2001	`.info` files
1.3	week 44 2001	database generation, sorting of similarity results
1.2	week 43 2001	fingerprints, metric, computation of fingerprint properties
1.1	10 Oct 2001	fragment class added, functional groups, generation of `.mol2` files, addition of hydrogens
1.0	week 40 2001	Basic DAIM