# Conformational Analysis Tools

## pre-release manual

# 1. Basic Principles

## 1.1 Analysing molecular archives - a quick overview

Conformational Analysis Tools (CAT) is controlled by an XML based script language called "CAT Script Language"(CSL). A basic CSL script to read a molecular archive and measure properties like RMSD, atom-atom distances or torsion angles looks like this:

```
<CSL id='analyse_archive‘ >
<!-- parameter section  -->
<read_parameters />
<md_parameters temperature='300' time_step_fs='1.0' history_start='0' history_freq='0.5' history_unit='ps' />
<read_template path='template_structure.pdb' />
<!-- wave definitions  -->
<wave_def value_type='RMSD' label='SystemRMSD'   note='root mean square displacement of complete system' />
<wave_def value_type='distance'  label='Neu3aGal2' note='H-H distance between axial H3 (Neu5Ac) and H2 (Gal)'
def_atoms='760 725'/>
<wave_def value_type='torsion' label='GLB_2OH'  note='2OH-torsion' def_atoms='1993 1992 1994 1995'/>
<!--  analysis -->
<analyse_archive  input_path='dyn_trj.xyz' />
<save_waves format='tsv' path='analysis_results'  />
<CSL />
```

The **<read_parameters** /> command reads general parameters like chemical elements from a file ‚CAT_par.xml', which has to be in the local or CAT home directory. **<md_parameters>** defines the simulation temperature and the history parameters which are used to assign the trajectory waves X scaling parameters (see waveform model for details). The **<read_template>** command is required to allocate memory for the molecular structure and to read and assign all the molecular properties like connectivities, H-bonds, residues and so on. The molecular parameters / properties which have to be analyzed by CAT are defined using the **<wave_def>** command. Each parameter (‚wave‘) is identified by a unique 'label'. The 'value_type' attribute determines the type of the property, the 'def_atoms' attribute contains a list of atoms IDs. The major command command in the script shown above is **<analyse_archive>** which defines the path and the format of the molecular archive and performs the analysis. **<save_waves>** finally saves all the results in the requested format to harddisk.

## 1.2 Installation

CAT is written in ANSI C and therefore should run on any system.Binary versions of the program are provided for download on the CAT Homepage (http://www.md-simulations.de/CAT/).

Since CAT is designed to be a high throughput analysis engine, the program works without any graphical interface, all the analysis is performed using a terminal or unix shell. Prior to its usage it is required to set the CAT_HOME environment variable (e.g. in your .cshrc file, for details see UNIX manuals)

```
setenv CAT_HOME /catpath/CAT2005                    (catpath = path to the CAT2005 folder)
```

To make work with CAT more convenient it is recommended to define an alias for CAT using

```
alias CAT '$CAT_HOME/bin/CAT2005.osx                (Mac OS X version)
```

To analyse a molecular archive  it is recommended to call CAT from the working directory (where the simulation data is stored). An analysis e.g. using the CSL script ‘analyse_archive.csl‘ can simply be performed by entering the following command in the unix terminal:

```
CAT analyse_archive
```

CAT looks for the CSL scripts first in the local directory and then in  $CAT_HOME/csl/.

## 1.3 CAT Script Language (CSL)

CAT commands are organized as ‚empty‘ XML tags. The  XML tag format is defined like this:

<**command** *attribute1*=‘value‘ *attribute2*=‘value‘ />

or

<**command**
*attribute1*=‘value‘
*attribute2*=‘value‘
/>

It is important that the tags start and end markers (‘<‘ and ‘/>‘) are present otherwise CAT cannot evaluate the command correctly. ‘<xxx> command‘ and ‘<xxx> tag‘ is used as synonyms in this manual, ‘<!--‘ and ‘-->‘ encloses a comment in the script, to desactivate a command temporarily use ’<#‘.

To make the usage of CSL scripts more efficient it is recommended to use variables in the scripts:

```
<CSL id='analyse_xyz_archive‘>
<!-- parameter section  -->
<read_parameters />
<md_parameters temperature='{VAR_1}' time_step_fs='1.0' history_start='0' history_freq='{VAR_2} history_unit='ps' />
<!-- CAT commands  -->
<read_template path='{VAR_3}' />
<!-- wave definitions  -->
<read_subscript path='wave_def' />
<analyse_archive  input_path='{VAR_3}' />
<save_waves format='tsv' path='analysis_results'  />
<CSL />
```

Assuming that the modified script above is stored as 'analyse_xyz_archive.csl' in the CAT home directory and the wave definitions are stored in the file 'wave_def.csl' in the working directory one can analyze the archive 'dyn_trj.xyz' by entering the following command in the unix terminal:

```
CAT analyse_xyz_archive 300 0.1 dyn_trj.xyz
```

The first parameter is the path of the CSL script[1]. As mentioned already CAT looks in two places for the CSL script: first in the current working directory and then in the CAT home directory.  If the CSL script requires script parameters they have to be input after the script path. The parameters are decoded inside the script as {VAR_1},{VAR_2},{VAR_3},... The same variable can be used in multiple locations in the script. If the number of submitted parameters does not fit to the script CAT stops with an error. If you want to use blanks in a parameter  you have to use quotes (e.g. for a sequence of atom IDs  “11 15 17 19“).

[1] If CAT is called without any parameters it starts in an ‚interactive mode‘ and will ask for the script. Script variables are not supported in ‚interactive mode‘ . Therefore the above script does not work in ‚interactive mode‘.

# 1.4 The waveform model of data

## 1.4.1 Introduction

CAT was originally designed to analyze molecular trajectory data derived from molecular dynamics simulations which typically consists of thousands of values measured at evenly spaced intervals of time. The uniform spacing of its values along an axis of time is typical for waveform data. Following this concept  the term 'wave' (short for 'waveform') is used to describe a CAT object that contains an array of numbers which have an important property called 'uniform spacing'[1]. In the current version of CAT a wave is a more complex data structure than simply an array of uniformly spaced numbers (for details see Fig. 1).

In the CAT terminology numbers which correspond to a sequence of 3D structures in a molecular archive are stored in a wave called '*trajectory wave*'. If the archive contains structures which are stored during a molecular dynamics simulation the X scaling (ΔX) of the wave corresponds to the sampling interval used in the simulation script. CAT automaticaly calculates basic statistics and a histogram of the trajectory wave data and stores the results 'inside' the wave as 'wave properties'.

A second important wave type used in CAT are called '*grid waves*'. Grid waves in general can have any dimensionality but three dimensional grids are sufficient to solve most of the analytical problems in conformational analysis. An example for a one dimensional grid wave is a simple histogram. The numbers (data values) stored in the wave correspond to the population in percent. The labels for the X axis are

```
typedef struct {

int wave_dim;                             /* 0=stream, >0=grid */
long N_data;                              /* number of data points in wave*/
char label[RCL_SMAX];                     /* wave label  */
char value_type[20];                      /* value type e.g. 'energy' */
char value_unit[20];                      /* value units e.g. 'kcal/mol' */
float running_value;                      /* actual value  */
float running_value_add[WADDMAX];         /* actual additional values  */
float template_value;                     /*  reference  value  of  template  structure  */float
constrain_min_value;                      /* range limit min value  */
float constrain_max_value;                /* range limit value  */
char note[50];                            /* wave note e.g. comment */
int N_rows, N_columns, N_layers;          /* number of rows(X),columns (Y),layers (Z)  */
int X_periodic, Y_periodic, Z_periodic;   /* periodic scale   eg. X_periodic=360 for torsions  */
float X0, DX, Y0, DY, Z0, DZ;             /* wave scaling */
char X_unit[20], Y_unit[20], Z_unit[20];  /* value units e.g. 'kcal/mol' */
float wave_stat[10];                      /* 0=number of data points used for statistics,
                                             1= min, 2=max, 3=mean, 4=stddev, 5=sum, 6=range,
                                             7=flexibility  */
float wave_histogram[101];                /* histogram of wave data  */float wave_hist_par[10];
                                          /* histogram parameters: number of points, start,
                                            interval, min, max */
char wave_hist_unit[20];                  /* value units e.g. 'percent' or 'counts' */
float *wave_data;                         /* wave data  */
float *wave_data_add[WADDMAX];            /* wave data additional columns*/
int output_flag;                          /* output of wave data in tables and graphs (1=yes)*/
int active_flag;                          /* use wave for special analysis */
..........
}STRUCT_WAVES;
```

**Fig. 1:** The C structure definition of a CAT wave object

[1] This is very similar to the 'wave concept' of our favorite graphical data analysis program 'Igor Pro' (www.wavemetrics.com).

calculated based on the waves scaling parameters using the expression X[i]=X0+ΔX*(i-1) (i=index of data value in the wave, X0=starting X value, ΔX=difference in X value from one point to the next). The scaling of three dimensional grid waves for example is determined by six scaling parameters (X0, ΔX, Y0, ΔY, Z0, ΔZ) respectively.

### 1.4.2 The <wave_def> command - overview

To manage and organize data in data structures called „waves“ is a central concept in CAT. The <wave_def> command creates a new wave and its attributes assign the waves properties. Use <wave_def> commands in CSL scripts after the <read_template> command only. A list of the major attributes of the <wave_def> command is listed in Table 1. Their usage will be discribed in the workshops in more detail.

**Table 1: attributes of the <wave_def> command**

| attribute name | description | remarks |
|---|---|---|
| label | assigns a name to the wave | has to be unique |
| value_type | determines the value type of the data stored in the wave. | energy, linkage, torsion, angle, distance, hbond, R_gyr, RMSD |
| value_unit | the unit of the data stored in the wave | ‚degrees‘, ‚kcal/mole‘, ... |
| note | wave description | user defined, not evaluated by CAT |
| def_atoms | atom IDs defining torsion, angle, distance, ... | can be ‘12 17‘ or ’1:GLY_2:O 1:ASN_3:C‘ |
| atom_type | atom types defining torsions, angles, distances, ... | ‘atom‘ (default); ‘pseudo_atom‘, ‘residue‘, ‘molecule‘ |
| wave_dimensions | number of rows, columns, layers | used for grid waves only |
| wave_scaling | start and delta value for each dimension | X0, DX, Y0, DY, Z0, DZ (trajectory waves: X0, DX = time, Y0, DY=value) |
| periodic_scale | periodic scale in x,y,z-dimension | used for torsions only |
| constraints | defines a min and max value | used for constraint analysis / minimization |
| output_flag | activate wave for output (save_wave, save_plot) | 1=output (default); 0= no output of wave |
| active_flag | set wave to 'active' | evaluated in some analysis functions |
| x_wave, y_wave, z_wave | defines wave used for x, y or z dimension | used for value_type='grid' only |
| grouping_type | defines grouping type | grid, wave |
| grouping_parameters | used for grouping | grid: interval, gap |
| groups_xml_file | path of groups wave; default = 'groups' | used for grouping_type='wave' only |
| map_wave | maps statistics (mean, min, max) of selected wave to grid | works with value_type='grid' and 'linkage' |
| molecule_id | used for value_type='ramachandran', determines which subset of atoms to use for phi/psi map | -1: activated atoms;<br>0:system; >0: molecule_id |
| level | used for value_type='RMSD'<br>determines ID_list level | molecule, residue, atom |
| ID_list | used for value_type='RMSD'; IDs of molecules, residues, atoms used for RMSD calculation | max. 20 IDs allowed |

### 1.4.3 The <wave_def> command - examples

#### Analyzing RMSD

One of the basic parameters which can be evaluated from molecular dynamics simulations is the root mean square displacement of atoms:

```
<wave_def value_type='RMSD'  label='RMSD_M1'  level='molecule' ID_list='1'  note='RMSD value of molecule 1' />
```

#### Analyzing atom-atom distances, bond angles and torsions

The basic format of the <wave_def> command to analyse geometrical properties is very simple:

```
<wave_def value_type='distance'  label='Neu3aGal2'  def_atoms='15 27' note='H-H distance axial H3 (Neu5Ac) to H2 (Gal)'  />
<wave_def value_type='angle' label='GLC2_COC' def_atoms='23 24 16' note='C-O-C angle of glycosidic bond'/>
<wave_def value_type='torsion'  label='GLB2_W' def_atoms='27 26 41 42'  note='omega torsion'  />
```

*Important:* Wave labels have to be unique in a CSL script. Their length is restricted to 50 characters but it is recommended to use shorter labels. If you want to add some comments or notes to the wave use the ‚note‘ attribute of the <wave_def> command. The ‘note‘ attribute to is not required but it is recommended for documentation purpose. The atoms defining a distance, angle or torsion are identified by their ID (atom index) in the template structure or an ‘atom identifier string‘.

‘atom identifier string‘ syntax:  *moleculenumber***:***residuetype_residuenumber***:***atomname* (eg. 1:GLC_3:C)

The torsion scale in CAT is defined to be [-120, 240] by default. If you want to change this for a torsion wave you can do this by adding the wave_scaling attribute to the <wave_def> command and set a new Y0:

```
<wave_def value_type='torsion'  label='GLB2_W' def_atoms='27 26 41 42'  wave_scaling='0 0 -180 0 0 0' note='omega torsion'  />
```

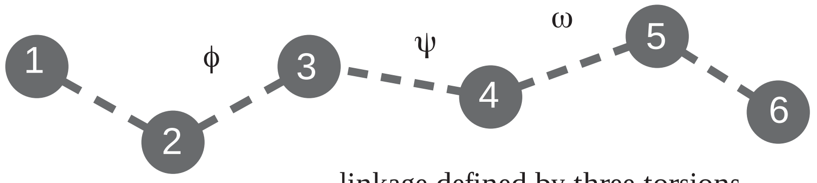
or if you want to change the torsion scale limit for all torsion waves use:

```
<analysis_parameters torsion_scale_start=‘-180‘  />
```

#### Analyzing linkages

CAT uses a special wave type for linkages. Linkages are defined by one to three torsions which means that linkage waves are torsion *grid waves*. Linkage waves are perfect to calculate histograms or energy maps of glycosidic bonds. 4 to 6 atom IDs are required to define one to three dimensional linkages.

```
<wave_def value_type='linkage' label='GLB2_14L' def_atoms='25 23 24 16 17'  note='1-4-linkage between GLB2 and GLC1' />
```



$\phi$ = 1-2-3-4
$\psi$ = 2-3-4-5
$\omega$ = 3-4-5-6

linkage defined by three torsions

#### Grouping values

In conformational analysis it is very important that one can treat ranges of values as groups. If you want to analyse the population distribution of ($sp^3$-$sp^3$)-rotamers it is necessary to separate the values of $\theta$ into three groups: A=[$-120\leq\theta<0$], B=[$0\leq\theta<120$] and C=[$120\leq\theta<240$]. This is very easy in CAT just type:

```
<wave_def value_type='torsion'  label='GLB2_W' def_atoms='27 26 41 42'  grouping_type='grid' grouping_parameters='120 0' />
```

The second parameter in grouping_parameters is the width of the gap (something like a 'transition state'). This will be explained in one of the workshops in more detail.

## 1.5 Supported molecular file formats

Molecular file formats which are currently supported by the <read_template> and <analyse_archive> commands are (most of them can be output as well):

| Format | Extensions | CAT Code | Comment |
|---|---|---|---|
| XMol | xyz | xyz | |
| PDB | pdb, pdbq, pqr | pdb, pdbq, pqr | special option for multi pdb files: pdbarc = no connectivities |
| DISCOVER | car,cor,arc, mdf | msi | connectivities are read from mdf file |
| SYBYL | mol2 | mol2 | |
| Maestro | mae | maestro | read only |
| MacroModel | dat, out | macromodel | read only |
| TINKER | tnk, tarc | tinker | |
| AMBER | prmtop, crd,crdbox | prmtop, crd,crdbox | |
| NAMD | psf | psf | |
| GROMACS | gro, itp | gro, itp | itp (connectivities and charges only, no #include molecules supported yet) |

## 1.6 Supported output file formats for results

| Format | Extensions | CAT Code | Comment |
|---|---|---|---|
| XML | xml | xml | CAT format (have a look in CAT_log.xml as well) |
| Igor | itx | igor | Igor text format (www.wavemetrics.com) |
| Table | tsv | tsv | TAB delimited table |
| Xmgrace | xvg | xvg | 1D data streams only |
| SVG | svg | svg | Scalable Vector Graphics |
| GAUSSIAN cube | cube | cube | used for density grids |

# 2. Analysis functions supporting NMR

## 2.1 Analyzing $^3$J vicinal coupling constants

```
<wave_def value_type='3J_coupling' label='GLC_H6R_H5_3J'  note='3J(H.H) coupling constant' def_atoms='11 8 9 10'
parameters='0.15 -0.96 7.49 0 0 0.15 0 7.49 0 0 ' />
```

```
<wave_def value_type='3J_coupling2' label='GLC_H6R_H5_3J'  note='3J(H.H) coupling constant' def_atoms='11 8 9 10'
parameters='0.15 -0.96 7.49 0 0' />
```

$$^3J = A + B\cos\varphi + C(\cos\varphi)^2 + D\sin\varphi + E(\sin\varphi)^2 \tag{1}$$

$$^3J = A + B\cos\varphi + C(\cos 2\varphi) + D\sin\varphi + E(\sin 2\varphi) \tag{2}$$

The $^3$J waves are similar to torsion waves except that extra parameters are required to calculate the vicinal coupling constant using equation 1 or 2. It is possible to define different parameter sets[1] for the torsion range [0,90] and [90,180] degrees.

## 2.2 Proton-Proton distance matrix

H-H distances of hydrogen atoms can be determined from NOE intensities. In order to analyze which H atoms are close in space it is possible to calculate a $r^{-6}$ weighted distance matrix from a MD trajectory:

```
<CSL id='HHdm'>
<!-- parameters section  -->
<read_parameters />
<read_color_table table_id='greenwhitered' />
<!-- analysis section  -->
<read_template path='mol.pdb'/>
<!--  activate all H-Atoms of molecule 1  -->
<activate_atoms mode='deactivate' />
<#activate_atoms  activate_type='molecule' activate_list='1' filter_type='element'  filter_def='H'  />
<activate_atoms  activate_type='molecule' activate_list='1' filter_type='atomtype'  filter_def='nonpolarH'  />
<plot_parameter width='400' height='400' fontsize='14' title='H-H Distance Matrix' x_label='H atoms' y_label='H atoms'
value_min='1' value_max='5'/>
<analyse_distance_matrix level='atom' input_format='xyz' input_path='dyn_trj.xyz'  output_path='HHdm' weighting='r^-6'  />
<CSL/>
```

[1] The parameters shown are just examples. Check literature for appropriate parameters to calculate coupling constants.

# 3. Analyzing hydrogen bonds

## 3.1 Hydrogen bonds trajectories and statistics

There are several functions implemented into CAT to analyse hydrogen bonds. One of them is to define waves with value_type='hbond' which results in a trajectory of the H-bond energy or ‘strength‘ (see below) of the H bond defined by three atoms (D-H-A).

The <wave_def> command offers three possibilities to define the atoms involved in the hydrogen bond:

```
<wave_def value_type='hbond' label='Interres_O3_O2'  def_atoms='16 17 28' />
<wave_def value_type='hbond' label='Interres_O3_O2'  def_atoms='1:GLC_1:O3 1:GLC_1:HO3  1:GLC_2:O2'  />
<wave_def value_type='hbond' label='HB_D16_17_A28' />
```

In the last version CAT decodes the atom IDs from the label string. It is a very quick and efficient way to access hbond trajectories since in some general H bond analysis functions CAT outputs in the logfile the H bonds found in the label format/name shown above.

To check H bonds of a solute atom to solvent molecules in general one can use

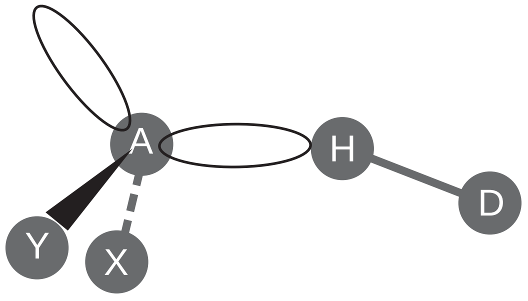
```
<wave_def value_type='hbond' label='HB_D16_17_solv' />
<wave_def value_type='hbond' label='HB_solv_A16'  mode='population' />
```

In the first example each solvent molecule is checked whether it forms a hydrogen bond with the donor atom 16 (solvent acts as the acceptor). In the second example the solvent is only taken into account when it acts as the donor in the hydrogen bond. The second example shows how to change the output mode, instead of the total hydrogen bond energy the number of hbonds formed with the solvent is output.

```
<analysis_parameters hbond_model='DH-A' hbond_max_distance='3.0' hbond_min_angle='120'
hbond_max_acc_orb_rms='30' />
```

There are three H bond models available in CAT (hbond_model='D-A', 'DH-A' , 'DH-AO'). The most simple model (D-A) checks only whether the donor-acceptor distance is smaller than *hbond_max_distance*.

In the frequently used ‚DH-A‘ model a hydrogen bond between a polar H-atom and an acceptor atom exists if the distance H-A is lower than *hbond_max_distance* and the D-H-A angle is greater than *hbond_min_angle*. A more advanced but still ‚experimental‘ model (DH-AO) evaluates the position of the H-atom with respect to the orientation of acceptor lone pair and a *hbond_max_acc_orb_rms* deviation is tolerated only. As a first approximation standard tetrahedral geometry is assumed for orbital orientation. The default values 3.0, 120, 30 (see above) can be changed using the <analysis_parameters> tag. It is recommended to use the 'DH-A' model at the moment.



Hydrogen bond parameters:

*hbond_max_distance* >= H-A

*hbond_min_angle* <= D-H-A

*hbond_max_acc_orb_rms* <= Acc_Orb_RMS

$\mathrm{Acc\_Orb\_RMS} = \mathrm{sqrt}(((X\text{-}A\text{-}H)-109)^2+((Y\text{-}A\text{-}H)-109)^2)$

The ‘strength‘ or energy of a hydrogen bond is defined as (usage of individual terms depends on the model used)

$$HB_S = \left(\frac{A}{d_{HA}^{12}} - \frac{\ddot{B}}{d_{HA}^{10}}\right)(-\cos\alpha)\frac{10}{\mathrm{Acc\_Orb\_RMS}} \qquad (3)$$

A=55332.873, B=18393.199 for N,O and A=298023.224, B=57220.459 for S as Acceptor[1]

*Important:* Several wave labels for H-bond waves are reserved for analyzing the sum of intra- and intermolecular hydrogen bonds, H-bonds to solvent and H-bonds of solvent molecules to other solvent molecules:

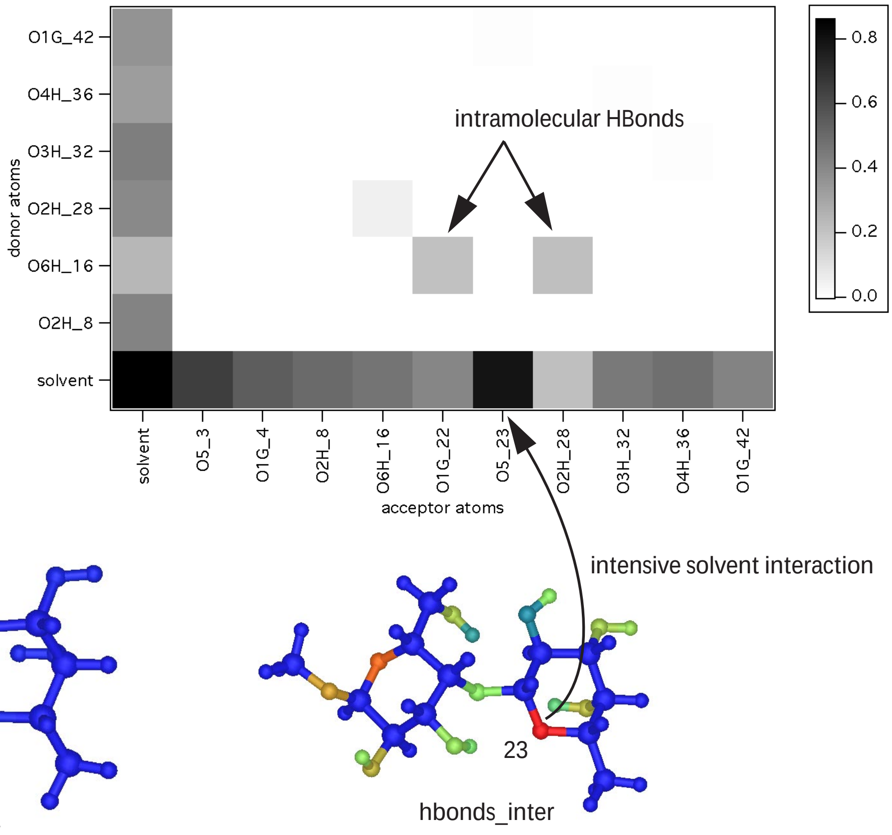
```
<wave_def value_type='hbond' label='Sys_hbonds_intra'  />
<wave_def value_type='hbond' label='Sys_hbonds_inter'  />
<wave_def value_type='hbond' label='Sys_hbonds_2solv'  />
<wave_def value_type='hbond' label='Sys_hbonds_solvent'  />
```

## 3.2 Hbond interaction matrix

A very efficient way to analyze the hydrogen bonding pattern of a molecular system is calculating an interaction matrix where the donor atoms are listed on one axis and the acceptor atom on the other. The propability of finding a hydrogen bond between donor-acceptor pairs is color coded in the plot.

```
<CSL id='analyse_Hbonds_matrix'>
<!-- parameters section  -->
<read_parameters />
<read_color_table  table_id='intensity'/>
<analysis_parameters hbond_max_distance='2.5' hbond_min_angle='120' hbond_max_acc_orb_rms='30'
solvent_analysis_level='sum' occupancy_mode='hbonds_intra' temp_fact_mode='hbonds_inter'/>
<!-- analysis section  -->
<read_template format='msi' path='{VAR_1}'/>
<plot_parameter width='400' height='400' fontsize='14' title='H Bonds Matrix'  value_min='0' value_max='1'/>
<analyse_hbonds_matrix input_format='xyz' input_path='{VAR_1}_trj'  output_path='{VAR_1}_HBmatrix' frames='1 1000 1' />
<save_template format='pdb' path='{VAR_1}_HBmapping'/>
<CSL/>
```

[1] Parameters from AutoDOCK 3.05 Manual



Setting the ‘occupancy_mode‘ and the ‘temp_fact_mode‘ to ‘hbonds_intra‘ or ‘hbonds_inter‘ will assign the propability to find an atom involved in hydrogen bonding to the occupancy / temperature factor fields of the molecular structure. Using the <save_template> command one can output these propabilities to a pdb file using the occupancy and beta value columns to store the values. Since some atoms that can serve as donors and acceptors the donor 'intensity' is assigned to the connected (polar) H atom and the acceptor propability to the acceptor atom.

# 4. Analyzing Solvent Effects

## 4.1 Radial Distribution Functions

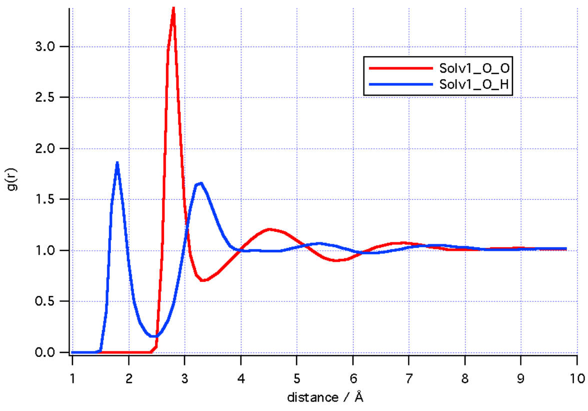
Radial distribution functions are calculated using equation 4.

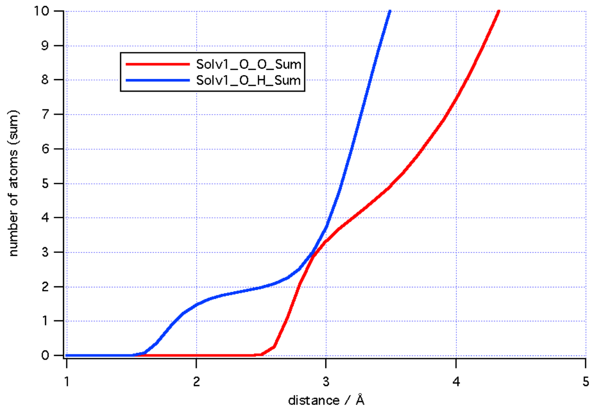
$$g_{AB}(r) = \frac{V}{4\pi r^2 \Delta r N_F N_A N_B} \sum_{n=1}^{N_F} \sum_{i=1}^{N_A} \sum_{j=1}^{N_B} Q_m(r; r_{A_iA_j}) \quad (4)$$

where $V$ is the total Volume (= $\frac{4}{3} \cdot \pi r_{max}^3$), $N_F$ is the total number of frames (snapshots), $N_A$ is the total number of center atoms A in the Volume, $N_B$ is the total number of atoms B respectively. $Q_n$ is the counting function for frames $n$. $Q$=1 if $r - \Delta r / 2 \le r_{A_iB_j} < r + (\Delta r / 2)$ otherwise $Q$=0.

```
<CSL id='analyse_rdf‘>
<!-- parameter section  -->
<read_parameters />
<plot_parameters width='600' height='400' fontsize='12' title='Radial Distribution Functions' x_label='Distance' y_label='rdf' />
<analysis_parameters solvent_residue1='SOL'  />
<!-- analysis section  -->
<read_template format='pdb' path='{VAR_1}' />
<analyse_rdf mode='solvent1'  center_element='O'  input_format='xyz' input_path='{VAR_1}_trj'  output_path='{VAR_1}_O_H_'
weighting='volume' search_element='H' search_molecule='solvent1'  search_radius='10' />
<analyse_rdf mode='solvent1'  center_element='O'  input_format='xyz' input_path='{VAR_1}_trj'  output_path='{VAR_1}_O_O_'
weighting='volume' search_element='O' search_molecule='solvent1'  search_radius='10' />
<CSL />
```

Currently the <analyse_rdf> command outputs the results in SVG and Igor format only.





CAT can handle solvent mixtures. Solvent molecules are detected by their residue name. You can define the residue name for solvent1 and solvent2 using:

```
<analysis_parameter solvent_residue1='SOL'   solvent_residue2='DMS' />
```

If you want to calculate radial distribution functions using explicit atoms as centers there are two possibilities:

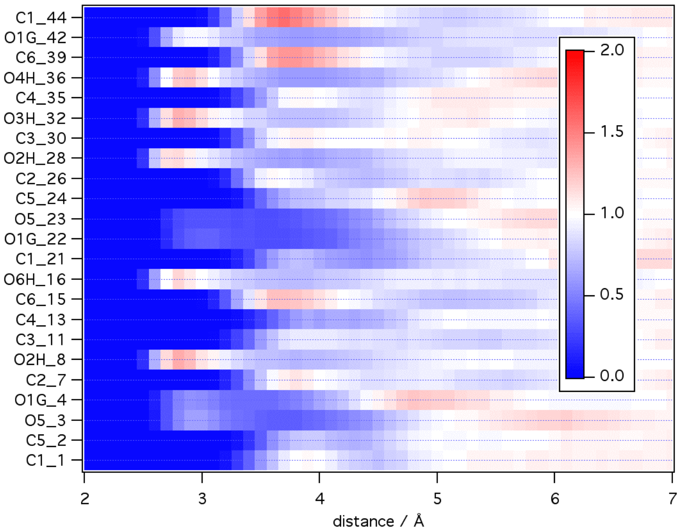
```
<analyse_rdf  center_id='20 25 31' input_format='xyz' input_path='{VAR_1}_trj'  output_path='{VAR_1}_M1_O_'
weighting='volume' search_element='O' search_molecule='solvent1'  search_radius='10' />
```

As a result you get a rdf profile for each center separatly.

An efficient method to define atom centers for rdf analysis is using the <activate_atoms> command.

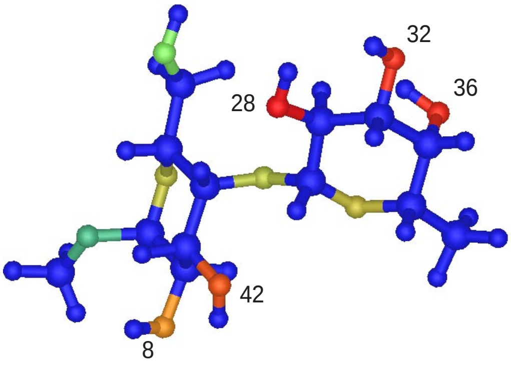
```
<CSL id='analyse_rdf‘>
<!-- parameter section  -->
<read_parameters />
<plot_parameter width='600' height='400' fontsize='12' title='Radial Distribution Functions' x_label='Distance' y_label='rdf' />
<analysis_parameters solvent_residue1='SOL'  />
<!-- analysis section -->
<read_template format='pdb' path='{VAR_1}' />
<activate_atoms mode='deactivate' />
<activate_atoms activate_type='molecule' activate_list='1' filter_type='atomtype' filter_def='heavy' />
<analyse_rdf  input_format='xyz' input_path='{VAR_1}_trj'  output_path='{VAR_1}_M1_O_' weighting='volume'
search_element='O' search_molecule='solvent1'  search_radius='10' />
<CSL />
```

The result of such an analysis can be visualized the best using an image plot representation.

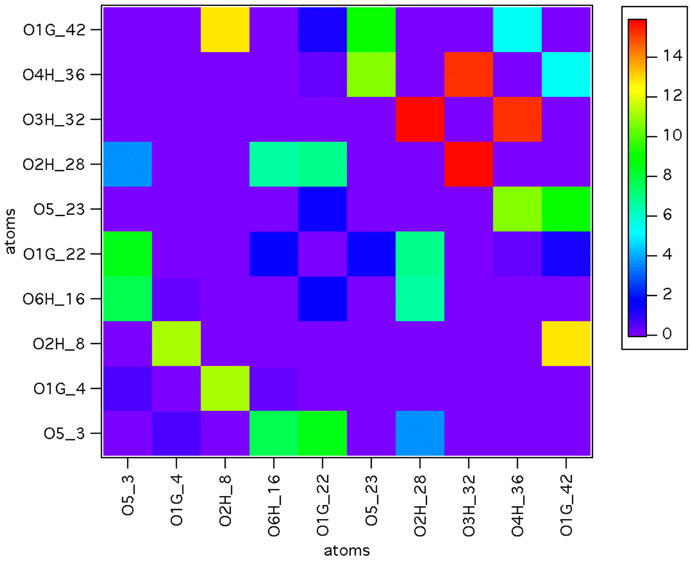


## 4.2 Detecting bridging water molecules

```
<CSL id='analyse_BW'>
<!-- parameter section  -->
<read_parameters />
<plot_parameter width='600' height='400' fontsize='12' title='Bridging water molecules' />
<analysis_parameters solvent_residue1='WTR'  />
<!-- analysis section -->
<read_template format='pdb' path='{VAR_1}' />
<!--  activate O atoms of molecule 1  -->
<activate_atoms mode='deactivate' />
<activate_atoms activate_type='molecule' activate_list='1' filter_type='element' filter_def='O' />
<analyse_neighbors mode='triple'  input_path='{VAR_1}_trj'  output_path='{VAR_1}_BW' search_radius='3.5' search_element='O'
search_molecule='solvent1'  frames='1 1000 1' />
<save_template format='pdb' path='{VAR_1}_BW'/>
<CSL />
```



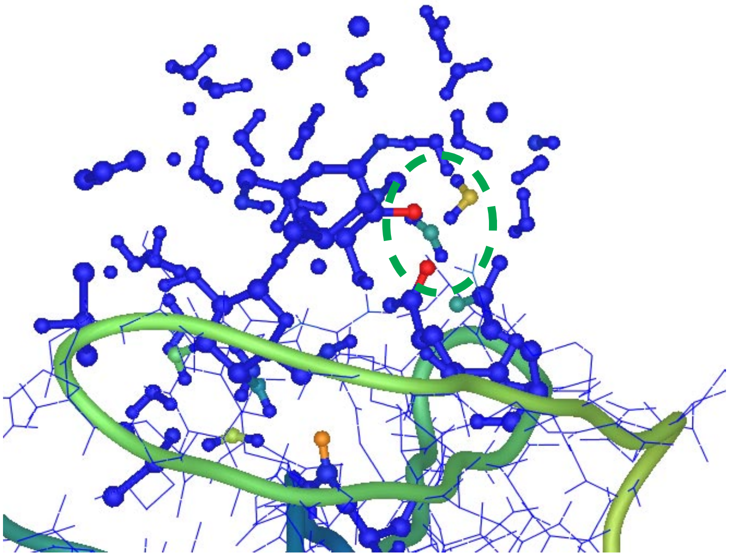
bridging waters



A more complex example where intensive use of the <activate_atoms> command has been made is the analysis of bridging waters in a protein-carbohydrate complex:

```
<CSL id='analyse_BW'>
<!-- parameter section  -->
<read_parameters />
<read_color_table  table_id='intensity'/>
<md_parameters temperature='300' time_step_fs='1.0' history_start='0' history_freq='0.5' history_unit='ps' />
<!-- analysis section  -->
<read_template format='pdb' path='{VAR_1}'/>
<activate_atoms mode='deactivate' />
<!--  activate O atoms of molecule 3  (ligand)-->
<activate_atoms mode='xlabel' activate_type='molecule' activate_list='3' filter_type='element' filter_def='O' />
<!--  activate binding Site of molecule 2  -->
<activate_atoms  mode='ylabel' center_type='molecule' center_list='3' sphere='5' activation_level='residue'
filter_type='molecule' filter_def='2'/>
<!--  deactivate solvent and H-Atoms  -->
<activate_atoms  mode='deactivate' activate_type='solvent1' />
<activate_atoms mode='deactivate' activate_type='molecule' activate_list='2' filter_type='element' filter_def='H' />
<plot_parameter width='400' height='400' fontsize='14' title='Bridging Waters'  value_min='0' value_max='1'/>
<analyse_neighbors mode='triple' input_format='xyz' stream_interval='5' cut_off='30' input_path='{VAR_1}_out'
output_path='{VAR_1}_N2D' search_radius='3.0' search_element='O' search_molecule='solvent'  />
<activate_atoms  activate_type='molecule' activate_list='2 3'  />
<activate_atoms center_type='molecule' center_list='3' sphere='5' activation_level='residue' />
<save_template format='pdb' path='{VAR_1}_BW' mode=‘active‘ />
<CSL/>
```

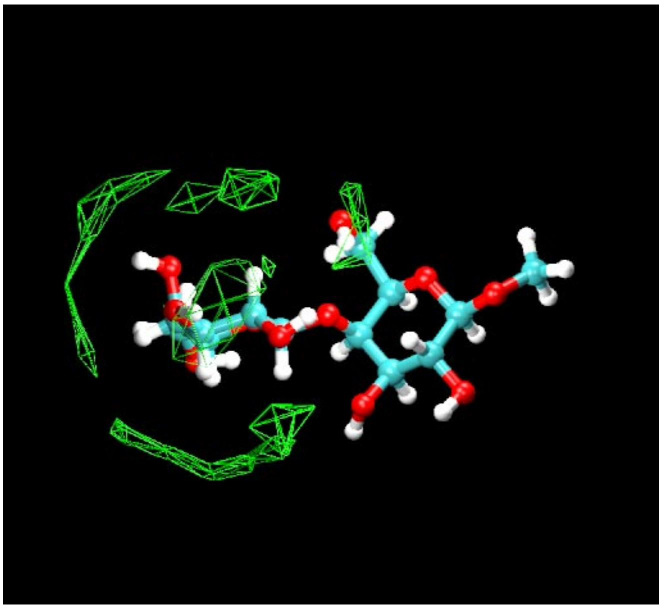
The vizualisation of the coded pdb file clearly demonstrates the involvment of bridging waters in binding.



## 4.3 Analysis of solvent densities in 3D

Tightly bound solvent molecules (e.g. on a protein surface or in the vicinity of a carbohydrate molecule) are restricted in their diffusion behavior. As a result the atom or particle density in particular areas in space are increased with respect to the bulk solvent. These higher densities can be detected (after proper orientation of the reference system) by averaging the population of volume elements (voxels) of a 3D grid over a MD run. The size of the grid is normally determined by the dimension of the solvent box.

The polulation density grid can be output in Gaussian Cube or XPLOR format, that can be visualized using program like VMD or Chimera.



The CSL script 'analyse_solvent3D' that performs such kind of analysis is shown on the next page.

CAT recognizes solvent molecules by their residue name (default: WTR HOH WAT SOL TIP). If the solvent has a different residue name <analysis_parameter solvent_residue1='MET'/> can be used to overwrite the default names[1].

<analyse_d3d> is the command that checks the extends of the solventbox, orientates the system, and determines the population densities relative to the average density (which is roughly the density of the bulk solvent if the box is sufficiently bigger than the solute dimensions).

After processing the trajectory the last frame is output without the solvent molecules as a reference frame[2] for display together with the 3D grid (D3D.cube) e.g. as isocontours using VMD.

The proper orientation of the system can easily be performed using the 3-atom orientation method of CAT (orientate_atoms='*a1 a2 a3*'), where atom *a1* is moved to the *origin*, *a2* aligned along the *x* axis and *a3* is orientated so that it becomes part of the *xy* plane. When moving the *a1* atom to the origin of the coordinate system the solvent is re-imaged, so that the box is symmetrical to the origin. Since CAT assumes that the solvent box has its unit vectors in x,y,z direction re-imaging can only be done if the box hasn't been rotated before. It is possible to advise CAT not to do the re-imaging of the solvent by setting <set_global_variables PBC_reimage_flag='off' />. It is also possible to set the grid dimensions explicitly using the grid_parameters attribute.

[1] In the script shown the command is deactivated using '<#'

[2] The conformation of this frame might not be representative for the whole ensemble

```
<CSL id='analyse_solvent3D'>
<!-- some parameters  -->
<read_parameters path='CAT_par.xml' />
<#analysis_parameter solvent_residue1='MET'  />
<!-- read template  -->
<read_template  path='template.pdb'  />
<!-- BOX dimensions, if not coded in template file  -->
<#analysis_parameter PBC='50 50 50' />
<#set_global_variables PBC_reimage_flag='off'  />
<!-- analyse trajectory -->
<analyse_d3d subunit='solvent' input_path='md_trj.xyz' search_element='O' orientate_atoms='27 26 28' grid_resolution='1.0'
grid_pa#rameters='-10  20 -10  10 -10  10' fr#ames='1 1000 1' />
<!-- output reference structure for display  -->
<activate_atoms mode='deactivate' />
<activate_atoms activate_type='molecule' activate_list='1'/>
<save_template format='pdb' path='D3Dref.pdb'   mode='active'/>
<!-- gaussian cube format for display with VMD  -->
<save_waves format='cube' path='D3D' wave_type='temp' />
<!-- XPLOR format for display with Chimera  -->
<#save_waves format='xplor' path='D3D' wave_type='temp' />
</CSL>
```

The script shows also examples how to deactivate complete commands (using <#) or attributes (inserting # in the attribute name)

# 5. High-throughput analysis of oligosaccharides using CAT

CAT was originally developed to perform conformational analysis of complex carbohydrates. Everybody working in this field knows that conformational analysis of carbohydrates differs significantly from protein analysis, which means that most of the analysis software (which was developed for proteins) is of only limited value to analyse a MD trajectory of a highly flexible branched oligosaccharide. in a waterbox.

## 5.1 Torsion angles

To describe stable conformations and the flexibility of a carbohydrate molecule usually torsion angles are used (particularly the glycosidic/exocyclic torsions φ/ψ/ω). Measuring the values of these torsion angles is usually the first step in conformational analysis of sugars. The bad news is that picking all the atoms that define the relevant torsions of a complex sugar by hand can be a tedious work, the good news is that CAT does the job automatically for standard sugars. CAT has some (limited) knowledge to determine a sugar automatically just from the 3D coordinates of the atoms (<find_sugar> command).

Let‘s have a closer look at the CSL script that outputs free energy maps[1] of glycosidic linkages, rotation profiles of exocyclic groups, flexibility measures and general statistics of the relevant torsions:

```
<CSL id='analyse_sugar_torsions'>
<read_parameters path='CAT_par.xml' />
<analysis_parameters torsion_scale_start='-120' torsion_scale_delta='10' />
<analysis_parameters  population_output='boltzmann'  rel_energy_cutoff='1.0' />
<!--  trajectory parameters  -->
<md_parameters temperature='300'  history_start='0' history_freq='1.0' history_unit='ps'/>
<!-- read template structure   -->
<read_template path='template.pdb'/>

<!-- auto define  sugar torsions and linkages     -->
<find_sugar />
<assign_torsions mode='wave_def'  def_atoms='sugar_omega'  IUPAC='xray' />
<assign_linkages mode='wave_def2'  wave_dim='2' def_atoms='sugar'  />
<assign_linkages mode='wave_def2'   wave_dim='3' def_atoms='sugar'  />
<assign_torsions mode='wave_def'  def_atoms='sugar_OH'  />
<assign_torsions mode='wave_def'  def_atoms='sugar_ring'  />

<!-- analyse archives   -->
<analyse_archive  input_path='md1nosA.pdb'   />
<analyse_archive  input_path='md1nosB.xyz'   />
<analyse_archive  input_path='md1nosC.xyz'   />

<!-- output  results   -->
<save_waves format='igor' path='torsions_results' />
<save_waves format='wave_def' path='torsions_def' />
<CSL/>
```

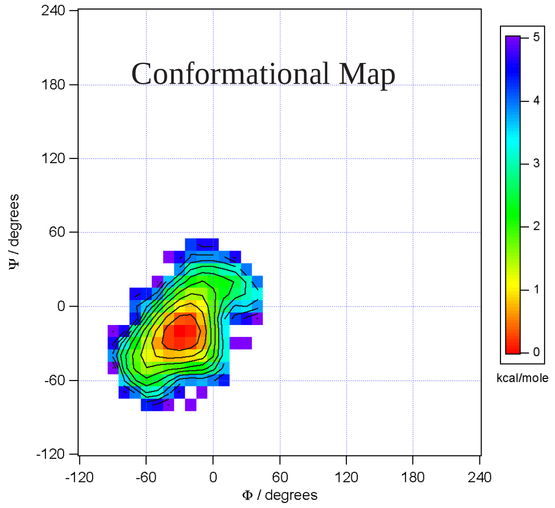
[1] Correct maps and profiles only if the conformational space has been sampled sufficiently (’conformational equilibrium’)

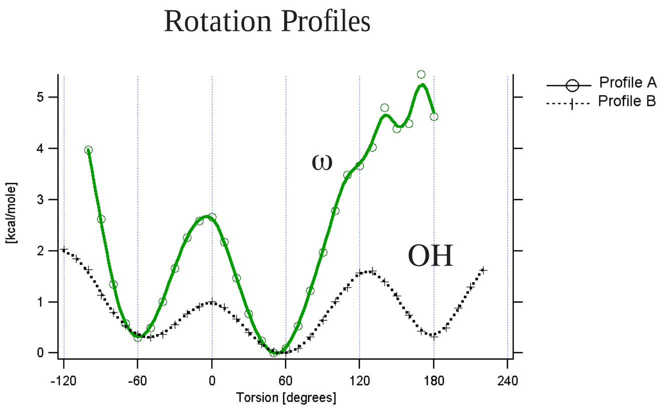
First general parameters are defined: The torsion scale starts at -120° and the profiles and maps have a resolution of 10°. The population_output=‘boltzmann‘ sets the flag to output free energy values for population statistics instead of percent or counts. The rel_energy_cutoff defines the cutoff for a simple flexibility analysis function: which area (in percent) of the map/profile can be occupied when the energy should be below *rel_energy_cutoff*. The temperature in <md_parameters> is required, otherwise the realtive population values cannot be transverted into energy values using the Boltzmann equation. The <read template> command assigns memory for the atoms and their properties and does some initial analysis on the structure[1]. The <find_sugar> command checks for carbohydrate rings in the template, assigns linkages paths for the individual branches of complex sugars, etc.

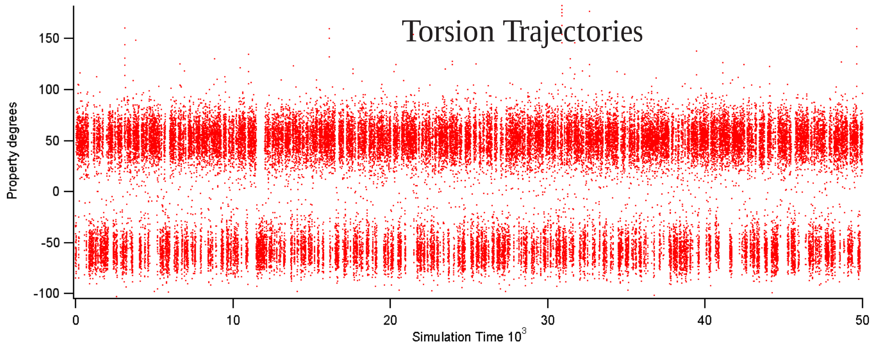
The <assign_torsions> and <assign_linkages> commands are not specific for carbohydrates, but if one inputs special keywords in ‘def_atoms‘ instead of atom identifiers, these commands can be used to create waves (mode=‘wave_def‘) that hold values for glycosidic torsions, conformational maps, exocyclic torsions, etc. Using wave_dim=‘3‘ in assign_linkages and def_atoms=‘sugar‘ defines all 1-6 linkages in the system.

Using the <analyse_archive> command(s) a molecular trajectory is processed sequentially frame-by-frame. Only the calculated values (e.g. torsions) are accumulated in memory (in the waves). Normally CAT does not read the whole archive into memory. CAT is developed to process large trajectories in background jobs. Finally the <save_waves> command outputs the results to hard disk. The option format=‘wave_def‘ can be used to create a CSL file with the <wave_def> command of all the waves in memory. This can be usefull for more advanced analysis using torsion waves.

Some results from a 50ns MD simulation of maltose (GLYCAM04) at 300K are shown in the figures.







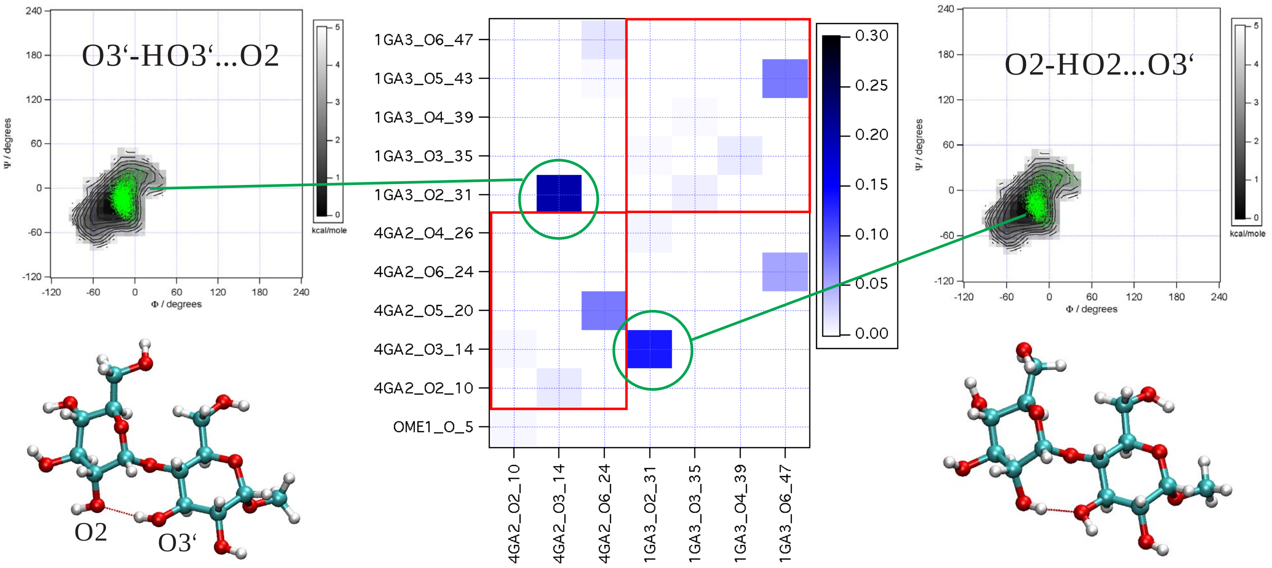
[1] Have a look at the CAT_log.xml file

## 5.2 Working with groups/constraints as filters in the analysis

The H-bond analysis of the trajectory of maltose reveals that there are two significant interresidual hydrogen bonds (O3‘-HO3‘...O2 and O2-HO2...O3‘). The two H-bonds obviously cannot be formed at the same time and the question may arise whether the selection which hydrogen bond is formed depends on the values of the ϕ/ψ torsions. To investigate this further ‘constraints‘ are included in the analysis:

```
<wave_def label='Phi'  value_type='torsion'  def_atoms='28 27 26 16' />
<wave_def label='Psi'  value_type='torsion'  def_atoms='27 26 16 17' />
<wave_def value_type='hbond' label='HBond'  def_atoms='1:4GA_2:O3 1:4GA_2:H3O 1:1GA_3:O2' constraints='-7 -4'  />
<analyse_archive  input_path='md1nosA.pdb'  extract_group='constraints'  />
```

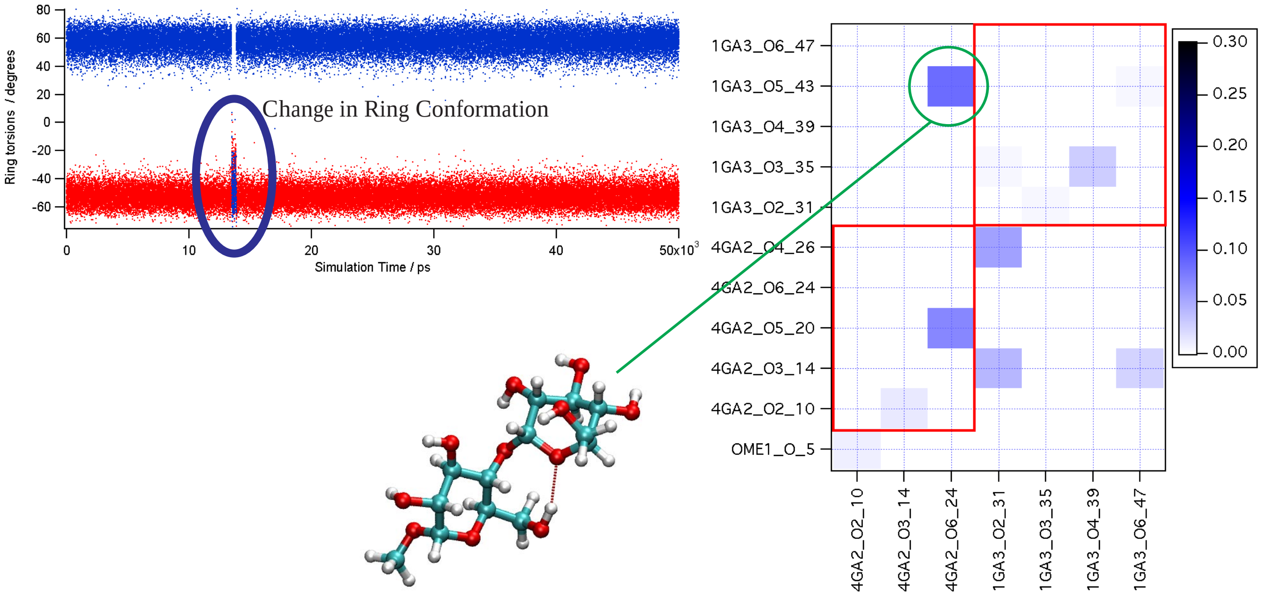
Using the option extract_group=‘constraints‘ in the <analyse_archive> command causes that analysis is performed only on those frames that fulfil all constraints previously defined. In the example shown on frames that have a strong hydrogen bond O3‘-HO3‘...O2 (energy between -7 and -4 kcal/mole). The ϕ/ψ torsions extracted using this filter are plotted together with the conformational maps derived from the complete analysis of the trajectory. The same has been performed using energy of the O2-HO2...O3‘ H-bond as a filter. The comparision of the plots reveals that both hydrogen bonds can be formed for a restricted range of ϕ/ψ combinations only and that there is no obvious difference in the ϕ/ψ values required to form both versions of the hydrogen bond between O2 and O3‘ since the extracted ϕ/ψ torsions populate the roughly same region of ϕ/ψ space in the map.



## 5.3 Extracting and analysing conformations from trajectories

Closely related to the ‘filtered analysis‘ described before is the strategy to extract conformations to a separate archive and analyse their properties individually. This strategy is very similiar to the analysis methods used in ‘experimental chemistry‘ where compounds are seperated e.g. using chromatography and each fraction is analyzed further individually as well.

In the trajectory plot of the ring torsions a conformational change in the (nonreducing) ring of maltose (50ns MD simulation, 300 K, GLYCAM04) could be observed (see figure). The goal is now to extract those frames with changed ring conformation and analyse their properties (e.g. hydrogen bonds).



Extraction of frames based on a property can be done using several methods in CAT. The very basic extracting method is to extract ranges of frames[1].

```
<convert_archive input_path='md1nosA.pdb'  output_path='extracted_frames.xyz'  frames='13500 13800 1' />
```

Frames 13500 to 13800 (interval=1) would we extracted to a new archive ‘extracted_frames.xyz‘.

Another possibility would be to define ‘constraints‘ and use the extract_group=‘constraints‘ option of the <convert_archive> command.

```
<wave_def label='Ring_tors1'  value_type='torsion'  def_atoms='37 41 43 27' constraints='-80 -20'  />

<convert_archive input_path='md1nosA.pdb'  output_path='extracted_frames.xyz'   extract_group='constraints'/>
```

This would be a more approbriate solution for the problem above since the property ‘torsion angle‘ is used to define a conformation instead of a time interval.

After extraction of all conformations with a twisted ring at the nonreducing end of the disaccharide to a new archive the hydrogen bonds are analyzed. It can be seen that the conformation might be stabilized by forming a hydrogen bond between O6‘ and O5.

This H-bond could not be detected by analysing the complete trajectory since the population of the conformations with a twisted ring is low in the ensemble and the H-bond doesn‘t show up as an intensive peak because most results are output as relative statistics. Even within the extracted ensemble the probability of finding the H-bond is only 15%.

Instead of extracting frames of low populated conformational families from a trajectory the method can be used to extract the highly populated conformational families and study their properties e.g. in comparision to experimental results.

[1] The ‘property‘ would be the time interval in this case.