ConfID

Conformational Identifier

for drug-like molecules

ConfID: an analytical method for conformational characterization of small molecules using molecular dynamics trajectories

Marcelo D. Polêto[a,b,*], *Bruno I.* Grisci[c,*], *Márcio* Dorn[c] *and Hugo* Verli[a]

[a]*Centro de Biotecnologia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 91509-900, Brazil.*
[b]*Departamento de Biologia Geral, Universidade Federal de Viçosa, Viçosa, 36570-000, Brazil.*
[c]*Institute of Informatics, Universidade Federal do Rio Grande do Sul, Porto Alegre, 91509-900, Brazil.*
[*]*These authors contributed equally to this work.*

August 15, 2020

Contents

CHAPTER 1

About

- This documentation refers to *ConfID* v. 1.2.1.
- The *ConfID* webpage: `http://sbcb.inf.ufrgs.br/confid`
- *ConfID* can be downloaded from the Snapcraft store: `https://snapcraft.io/confid`
- The complete source code is freely available at the repository `https://github.com/sbcblab/confid`
- *ConfID* is registered at Instituto Nacional da Propriedade Industrial (INPI) under the number BR512019001928-8 and is freely available under the license LGPL-3.0.

For the application note on *ConfID* published in Bioinformatics, please refer to:
Marcelo D. Polêto, Bruno I. Grisci, Marcio Dorn, Hugo Verli. *ConfID: an analytical method for conformational characterization of small molecules using molecular dynamics trajectories*, Bioinformatics, **2020**, btaa130, DOI: https://doi.org/10.1093/bioinformatics/btaa130

CHAPTER 2

ConfID Approach

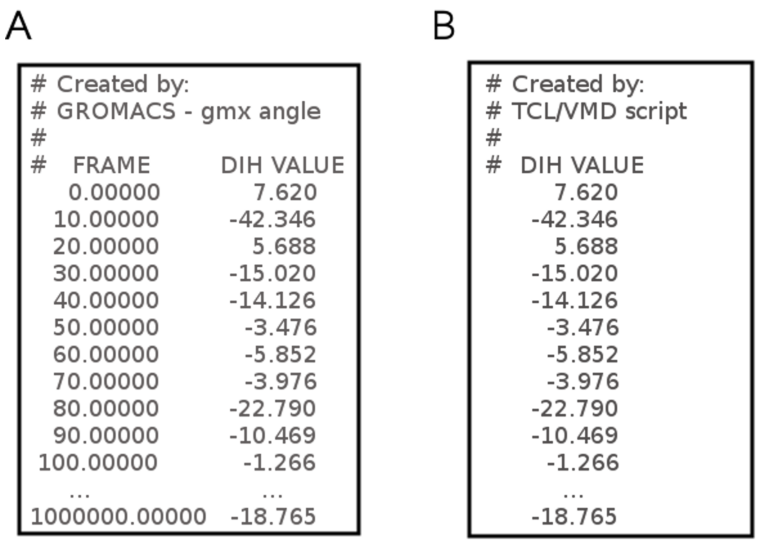
In recent decades, genetic algorithms and knowledge-based approaches have been employed to study the molecular flexibility of drug-like compounds [1, 2]. However, these methods are usually based on crystallographic information, and their calculations are made either in vacuum or with implicit solvent, which does not take into account the influence of explicit solvent dynamics on ligand conformational preferences [3, 4]. On the other hand, molecular dynamics simulations can be employed to grasp information regarding the conformational ensemble of drug-like molecules [5, 6, 7, 8, 9, 10]. However, due to the overwhelming amount of conformations generated, proper conformational population characterization is often relegated to clusterization algorithms, which can be insensitive to small conformational changes or tackled by manual efforts, which can be a source of errors. In this sense, we present *ConfID*: a tool for conformational characterization of drug-like molecules based on molecular dynamics trajectories.
ConfID is based on dihedral values of relevant torsions of drug-like compounds sampled throughout molecular dynamics (MD) simulations. By using such data, *ConfID* can identify conformational populations sampled and quantify their relative abundance while harnessing the benefits of MD simulations (explicit solvent and kinetics) and calculating time-dependent properties of each conformational population.

2.1 Basic Required Files

To properly characterize conformers of a drug-like molecule, *ConfID* requires only one file for each relevant torsion, containing their dihedral values sampled throughout MD simulations. Most common MD suites easily generate such files and our software was designed to be universally compatible.

For this, *ConfID* accepts two input formats: files containing both frame and respective dihedral values (2-columns) or 1-column file containing only dihedral values (Figure S2.1). For example: GROMACS tool [11] *gmx angle* outputs dihedral values evolution as a function of

time in a 2-column file type by default (Figure S2.1A), while VMD/NAMD suites [12, 13] can output dihedral values in a single column file (Figure S2.1B) through a TCL command-line. In either cases *ConfID* will generate dihedral distributions based on each file to identify dihedral populations sampled throughout the simulation.
It is crucial to notice that, if the format of files with dihedral values used as input does not contain any information regarding simulation time (1-column format), the total simulation time should be explicitly stated in the "config" input file (for more information, see Section 4.2). Besides that, all these files should have the exact same number of frames.



```
# Created by:
# GROMACS - gmx angle
#
#   FRAME          DIH VALUE
    0.00000          7.620
   10.00000        -42.346
   20.00000          5.688
   30.00000        -15.020
   40.00000        -14.126
   50.00000         -3.476
   60.00000         -5.852
   70.00000         -3.976
   80.00000        -22.790
   90.00000        -10.469
  100.00000         -1.266
     ...             ...
1000000.00000      -18.765
```

```
# Created by:
# TCL/VMD script
#
#  DIH VALUE
     7.620
   -42.346
     5.688
   -15.020
   -14.126
    -3.476
    -5.852
    -3.976
   -22.790
   -10.469
    -1.266
     ...
   -18.765
```

Figure 2.1: Input formats supported by *ConfID*. Depending on the MD simulation package used, dihedral values can be written in files accompanied by their respective frames (A) or not (B), but both formats are accepted. Lines starting with # or @ are ignored.

2.2 Definition of Dihedral Population

By reading the input files, *ConfID* generates dihedral distributions for each torsion, which are then used to identify dihedral populations. These distributions are smoothed using the Hann function [14] with a sliding window, obtaining a curve with a well-behaved gradient. Then, each dihedral population is assigned to a peak in the distribution (an angle with a larger distribution value than the values of its immediate neighbors) and flanked by valley regions (angles with distribution values below a given threshold that indicates a distribution so low that the angle should be considered spurious, or with a lower distribution than the values of their immediate neighbors).

This definition of dihedral populations (a distribution flanked by two valley regions and represented by its peak) produces a unique angle-population relationship and, therefore, a given dihedral value can only belong to a single dihedral population. Since each torsion may contain a variable number of dihedral populations, *ConfID* always provides all-region definitions (e.g. peaks and valley regions) as .xvg files that can be visualized by Xmgrace (by default, on folder

"Dihedral_Regions") for the user to check whether the criteria chosen for peak and valley assignment were accurate enough, as represented by Figure S2.2. It is also important to mention that users may tweak threshold values for peak and valley regions assignment so that it can be more or less rigorous, accordingly.

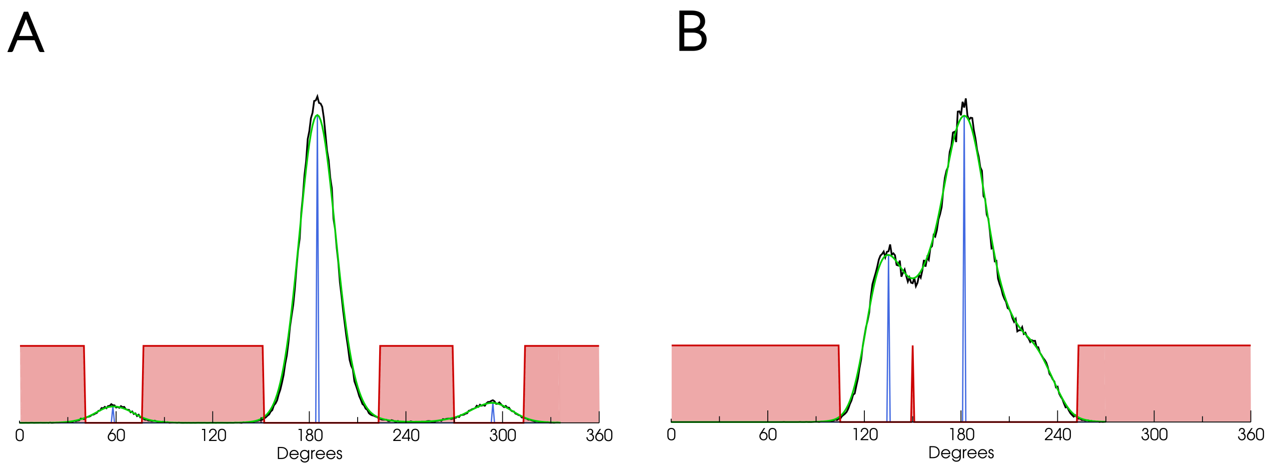


Figure 2.2: Dihedral populations and regions, as identified by *ConfID*, for well-defined dihedral populations (A) and overlapping dihedral populations (B). Original curves are shown in black, smoothed distributions are shown in green, identified peaks are shown in blue and valley regions are shown in red.

2.3 Definition of a Conformational Population

After identifying all possible dihedral populations, *ConfID* quantifies the frequency of all different tuples of dihedral populations over a simulation by evaluating the array of all dihedral values through time. Hence, a conformational population (conformer) is defined as a unique tuple of dihedral populations of the analyzed torsions of a molecule, i.e., a set of conformations sharing similar values for their respective dihedral angles.

Due to the usual picosecond MD output frequency, it is possible to encounter dihedral values outside any identified dihedral population. Such values are considered spurious due to their low frequency. Therefore, noisy conformations are defined as a given frame with a conformation with any torsional angle outside the dihedral population regions previously identified (i.e., angles within a valley region). These noisy conformations are assigned to a "Z-region" and are not considered by default for conformational population calculations.

In this sense, *ConfID* accuracy can be defined as $Accur\% = \frac{(Frames_{Total} - Frames_{Z-region}) \times 100}{Frames_{Total}}$. Our approach was thoroughly tested for a variety of ligands with different degrees of torsional freedom, and *ConfID* was able to address up to 98% of the conformations sampled in trajectories with well-defined dihedral populations (i.e., with no overlapping distributions). In cases in which the limit between dihedral populations was poorly-defined due to high flexibility, our method was still able to determine 84% of sampled conformations as its most poor performance. As a drawback, *ConfID* performs poorly when characterizing highly flexible molecules that present dihedral distributions with no peaks (homogeneous or almost flat distributions) and we advise that such cases must be taken with caution since they are still challenging for *ConfID* and other conformational characterization methods [1, 15, 16].

As mentioned before, *ConfID* accuracy is directly impacted by the correct assignment of peaks and valley regions, and we encourage users to test different criteria to guarantee the most precise characterization.

CHAPTER 3

ConfID Features

ConfID was designed to provide much useful structural information regarding drug-like molecules dynamics and conformational space. Below, we list *ConfID* features and possible applications of such information in research.

3.1 Conformational Characterization

The most straightforward information *ConfID* was designed to provide is a complete identification of all conformational populations sampled in an MD trajectory. The software provides a list with all populations detected, their composing dihedral populations (identified by their peak), and the conformational population relative frequency throughout the entire MD trajectory. Such information can be useful to characterize conformers of a drug-like molecule in different conditions, as exemplified in Figure S3.1. For transparency, *ConfID* also provides its accuracy value and Z-region size.

3.2 Structural Characterization of Conformational Populations

By default, *ConfID* generates, for each conformational population, a list of frames in which that conformation occurs, allowing for the user to dump these frames and further investigate possible chemical events (such as intramolecular and intermolecular interactions, as well as solvent rearrangement) that might explain the abundance of each conformer in solution. This information can be advantageously used to study NMR-NOESY data from a conformational perspective.

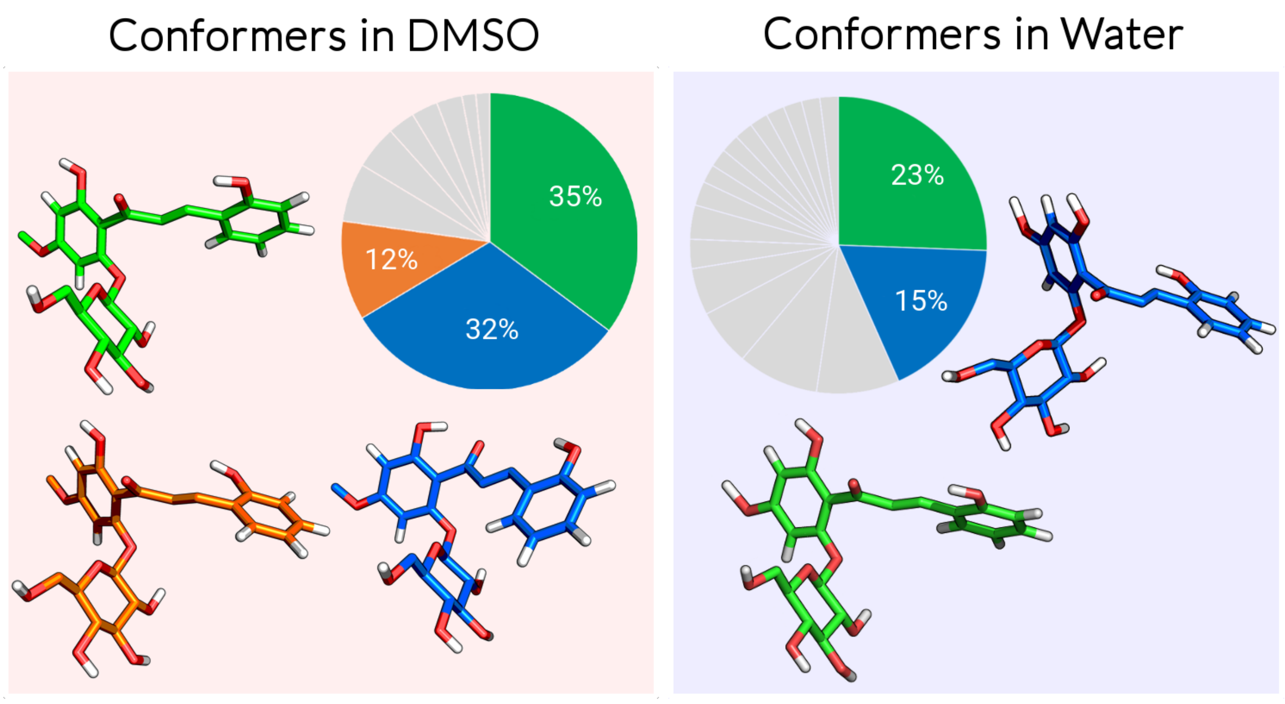


Figure 3.1: Characterization of conformational populations of a glycosilated chalcone simulated in different solvents [6] and relative abundance of each conformer, calculated by *ConfID*. Only conformers with relative abundance over 10% are shown.

3.3 Convergence of Conformational Sampling

One major challenge about MD studies on conformational sampling is the amount of simulation time required to sample the major representative states of a given molecule accurately. Due to the inherently fewer conformational degrees of freedom of drug-like molecules, simulation of such cases is usually on the few nanoseconds timescale [7, 9, 8]. To tackle this subject in a objective manner, *ConfID* tracks the evolution of unique sampled conformational populations throughout time and plots them as a function of time along with their relative frequency, providing robust information for the user to judge whether the simulation time used was sufficiently long or even to be used as a convergence criteria in automated protocols. Figure S3.2 is an example of such analyses, in which it is possible to assess convergence of conformational sampling by evaluating the complete conformational ensemble (Figure S3.2A) or by evaluating the relative abundance of each conformer (S3.2B).

3.4 Transition Events

In order to take advantage of MD kinetics information, *ConfID* was designed to track transition events between different conformational populations and single dihedral populations. Such transitions are quantified and plotted as graphs, in which nodes represent a unique conformational population, and edges represent transitions between nodes, as exemplified by Figure 5.1. For convenience, *ConfID* save all transition information in GML format by default, and users can evaluate these graphs in third-party software, such as Cytoscape [17]. Also, *ConfID* can

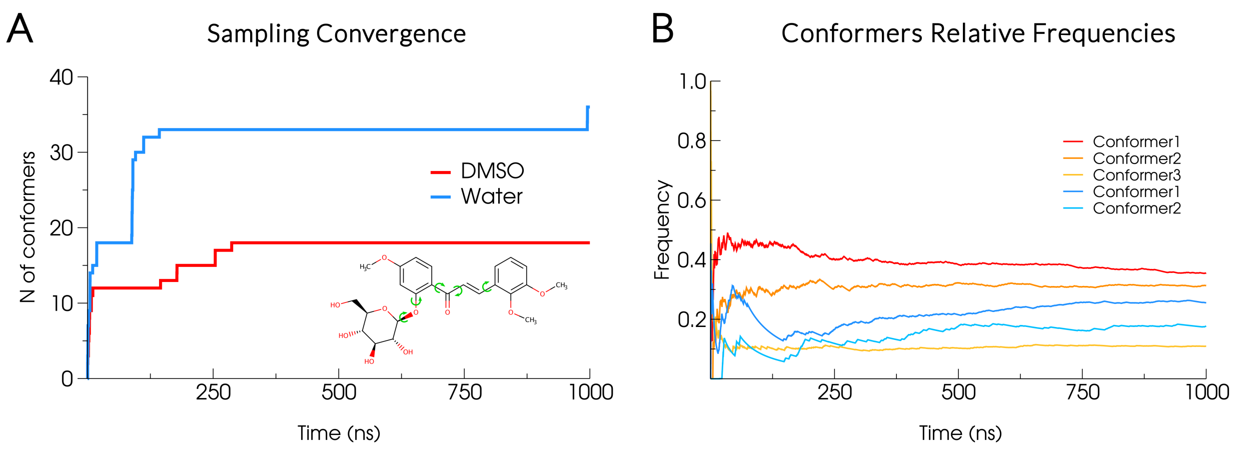


Figure 3.2: Conformational sampling characterized by *ConfID* for a glycosilated chalcone in organic (DMSO) and water solvent, showing that solvent chemical nature directly impact conformational sampling (A) and the relative frequencies of each conformational population (B). Curves in cold pallete represent data from simulations in water, while curves in warm pallete represent data from simulations in DMSO (only conformers with relative abundance over 10% are shown).

roughly plot such graphs by enabling this option on a configuration file (by default, it is disabled).

3.5 Time-dependent properties of conformational population

By tracking conformational populations throughout MD trajectory, *ConfID* calculates time-dependent properties of each conformational population. As mentioned above, *ConfID* also tracks transition events between dihedral populations, which allows the calculation of time-dependent properties for dihedral populations as well. Such properties are provided to the user as both lists and correlation graphs (Figure S3.4) and a pair of properties must be provided at the "config" file (See section 4.2). The functions available for time-dependent properties calculation are:

- *sum*: total time in a population
- *max*: maximum time spent in a population without leaving
- *min*: minimum time spent in a population without leaving
- *aver*: average time spent in a population without leaving
- *std*: standard deviation of the average time
- *median*: median time spent in a population without leaving
- *count*: the amount of times of a transition event entered this population

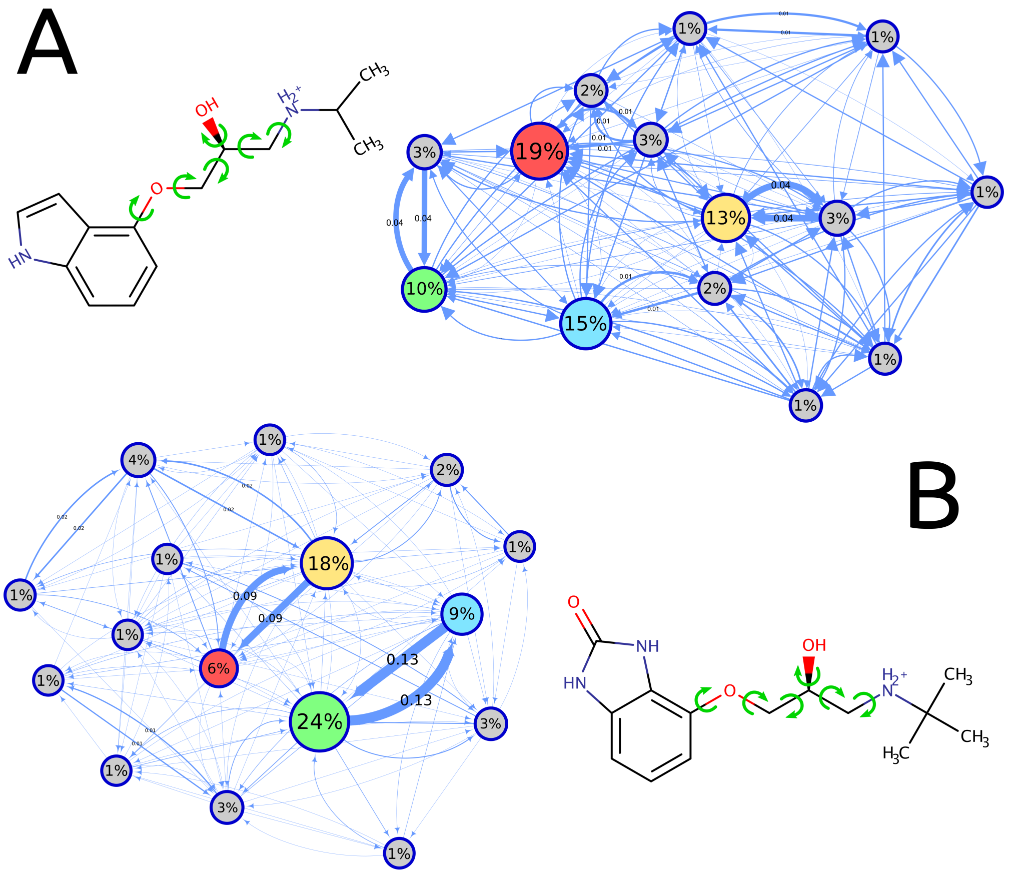


Figure 3.3: Conformational ensembles of beta-adrenergic antagonists (A- Pindolol and B- CGP-12177) considering 6 torsions (green arrows) shown as graphs and obtained by *ConfID* after convergence of explicit solvent MD simulations. Nodes with the same coloring represent the same conformational state for both molecules. Despite molecular resemblance, small chemical modifications in CGP-12177 yield a clear increase of the relative abundance of 3rd and 4th more abundant conformers of Pindolol (manuscript in preparation).

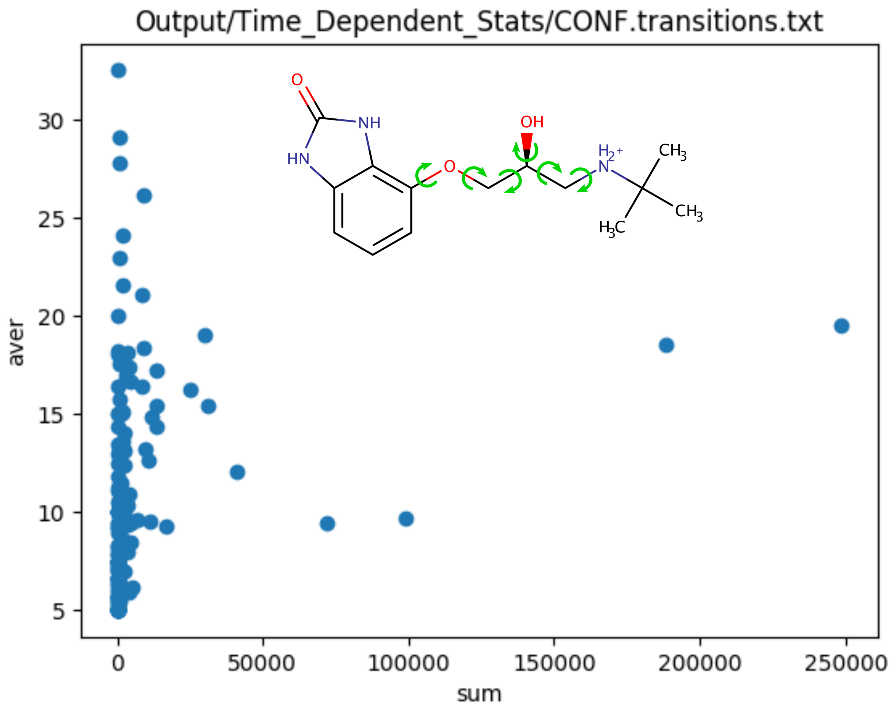


Figure 3.4: Graph of correlation between time-dependent properties *sum* and *aver* of the synthetic beta-adrenergic antagonist CGP-12177. Most abundant conformations (with higher *sum* values) do not necessarily have higher average times of existence.

CHAPTER 4

ConfID Inputs and Parameters

4.1 Input file

ConfID requires a "input.inp" file that should follow this template:

```
# Comment that is ignored
Path/to/file/file1.dat
Path/to/file/file2.dat
Path/to/file/fileN.dat
```

The "fileX.dat" files are the files containing the dihedral values sampled during MD simulations, as shown in Fig. 2.1. Each line of "input.inp" should contain the path to each file that should be analyzed by *ConfID*. The name and order of such files will be used by *ConfID* to identify the torsion angles.

4.2 Configuration file

To take full advantage of *ConfID* features, an optional configuration file ("config") can be used to set *ConfID* parameters, like the example below:

```
RESULTS_FOLDER          (string)          defaults to Output/Populations/
DIH_POP_FOLDER          (string)          defaults to Output/Dihedral_Regions/
NETWORK_FOLDER          (string)          defaults to Output/Networks/
TIME_STATS_FOLDER       (string)          defaults to Output/Time_Dependent_Stats/
SIM_TIME                (None or float)   [None / > 0.0] defaults to None
SHOW_Z                  (string)          [False / True] defaults to False
NETWORK_CUTOFF          (float)           [>= 0.0]  defaults to 0.01
PLOT_NETWORK            (string)          [False / True] defaults to False
CONVERGENCE_CUTOFF      (float)           [>= 0.0] defaults to 0.01
WINDOW_LEN              (integer)         [>=3, < 180, odd] defaults to 21
WINDOW                  (string)    [flat/hanning/hamming/bartlett/blackman] defaults to hanning
FACTOR_PEAK             (float)           [>= 1.0 and < FACTOR_VALLEY] defaults to 50.0
FACTOR_VALLEY           (float)           [>= 1.0 and > FACTOR_PEAK] defaults to 60.0
TIME_DEPENDENT_STATS    (string)          [False / True] defaults to True
DATA_1                  (string)          [sum/max/min/aver/std/median/count] defaults to sum
DATA_2                  (string)          [sum/max/min/aver/std/median/count] defaults to aver
```

The final file should look something like this:

```
RESULTS_FOLDER          Output/Populations/
DIH_POP_FOLDER          Output/Dihedral_Regions/
NETWORK_FOLDER          Output/Networks/
TIME_STATS_FOLDER       Output/Time_Dependent_Stats/
SIM_TIME                None
SHOW_Z                  False
NETWORK_CUTOFF          0.01
PLOT_NETWORK            False
CONVERGENCE_CUTOFF      0.01
WINDOW_LEN              21
WINDOW                  hanning
FACTOR_PEAK             50.0
FACTOR_VALLEY           60.0
TIME_DEPENDENT_STATS    True
DATA_1                  sum
DATA_2                  aver
```

Note that the values above are the actual default values that will be used if no "config" file is provided when running *ConfID*.
Following, we provide a brief explanation of each parameter available for the configuration file:

- RESULTS_FOLDER: specifies the directory in which output files should be saved.
- DIH_POP_FOLDER: specifies the directory in which output .xvg files should be saved.
- NETWORK_FOLDER: specifies the directory in which output network files should be saved.
- TIME_STATS_FOLDER: specifies the directory in which output transition files should be saved.
- SIM_TIME: specifies the total simulation time. SIM_TIME is mandatory if you are working with dihedral files with only one column (dihedral angle). Otherwise, it is optional but must be equal to the actual total simulation time. Must be a value larger than zero. It is important to notice that the timescale chosen here is the same for the time-dependent properties. We suggest the use of a picosecond timescale.
- SHOW_Z: a flag that determines if spurious dihedral regions (Z-region) should be presented in the results. They will be used in the internal calculations nevertheless. If this is True, please consider setting PLOT_NETWORK to False, as plotting the chart may become too slow.
- NETWORK_CUTOFF: the smallest transition frequency required for an edge to appear in the networks. If equal to 0.0, all edges are considered. If this cutoff is too small, please consider setting PLOT_NETWORK to False, as plotting the chart may become too slow.
- PLOT_NETWORK: if True, network figures for the transitions will be created using the Graphviz library. Network text files will be created if it is either True or False.
- CONVERGENCE_CUTOFF: the smallest population frequency at the end of the simulation required for the convergence file of that population to be generated. If equal to 0.0, all populations will be represented, but for a large number of dihedral angles, this can take a while.

- WINDOW_LEN: range of neighbouring angles in the original distribution that will be averaged (using the function defined in WINDOW) to smooth the distribution curve of each angle. Must be an odd integer between 3 and 180. The default is 21, which means the previous 10 and next 10 angles will be considered in the smooth of the distribution. This parameter is only available from *ConfID* version 1.2.1.

- WINDOW: the name of the function to average the angles distribution in the window range defined in WINDOW_LEN. For details about the functions see the SciPy reference: https://docs.scipy.org/doc/numpy/reference/routines.window.html. The default is 'hanning'. The available functions are 'flat', 'hanning', 'hamming', 'bartlett', and 'blackman'. This parameter is only available from *ConfID* version 1.2.1.

- FACTOR_PEAK: a factor that sets the constriction for peaks selection. Larger values lessen the constriction.

- FACTOR_VALLEY: a factor that sets the constriction for valleys selection. Lower values lessen the constriction.

- TIME_DEPENDENT_STATS: flag that determines if the statistics of the time stayed at each population should be computed.

- DATA_1: list of functions that should be used as the x-axis of the charts of the statistics of the time stayed at each population and how the report should be ordered.

- DATA_2: list of functions that should be used as the y-axis of the charts of the statistics of the time stayed at each population and how the report should be ordered.

CHAPTER 5

ConfID Outputs

Here we describe all outputs generated by *ConfID*. File names are colored in **blue**, while folder names are colored in **green**. By default, all outputs in *.xvg files are graphs and can be visualized in the freely available *Xmgrace* software.

ConfID create a folder **Output** with four new folders named, by default, **Dihedral_Regions**, **Populations**, **Networks** and **Time_Dependent_Stats**.

5.1 Dihedral_Regions

This folder contains information regarding dihedral distributions and their regions (peaks and valleys) used by *ConfID* to determine dihedral populations for input torsions.

5.1.1 Dihedrals distributions

By counting the frequency of dihedral values provided as input, *ConfID* calculates a distribution profile for each torsion (**.dist.xvg*), as shown below.

Dihedral Distribution

```
# Created by:
#       ConfID - Conformational Identifier
@    title "Dihedral Distribution: DIH1"
@    xaxis label "Degrees"
@    yaxis label "Distribution"
@TYPE xy
@    subtitle "average angle: 26.0301\So\N"
      -93     0.000000
      -92     0.000010
      -91     0.000000
      -90     0.000010
      -89     0.000000
      -88     0.000000
      -87     0.000000
      -86     0.000010
      ...
```

5.1.2 Dihedrals populations

After calculating the dihedral distribution of each torsion, *ConfID* smooths the distribution by using the Hann function with a sliding window (**.distsmooth.xvg*), identifies peaks (**.peaks.xvg* and **.distpeaks.xvg*) and valleys (**.distvalleys.xvg*). If needed, *ConfID* also treats periodicity by simply shifting dihedral values by 180° (**.distshift.xvg*). An example of dihedral population idenfitication can be seen in Figure 2.2.

5.2 Populations

This folder contains information regarding conformational populations identified and their relative abundance.

5.2.1 Conformational populations

A summary of identified dihedral regions is also available at *DIHEDRAL_REGIONS.txt*, as shown below.

DIHEDRAL_REGIONS.txt

```
Angle #1:   Dihedrals/DIH1.dist.xvg
[(-25, 74)                  # count:   99663 # freq:   0.997 # peak:    27 # mean:    26.100 #
    std:  16.795 # median:    26.270]
Angle #2:   Dihedrals/DIH2.dist.xvg
[(-156, -126)               # count:    7815 # freq:   0.078 # peak:  -129 # mean:  -136.652 #
    std:   7.015 # median:  -135.676,
 (-126, -39)                # count:   91970 # freq:   0.920 # peak:   -89 # mean:   -88.347 #
     std:  16.087 # median:   -88.995]
Angle #3:   Dihedrals/DIH3.dist.xvg
[(-79, 44)                  # count:   53317 # freq:   0.533 # peak:   -10 # mean:    -9.886 #
    std:  21.683 # median:    -9.963,
 (44, 136)                  # count:   46438 # freq:   0.464 # peak:    94 # mean:    90.473 #
     std:  16.772 # median:    91.737]
Angle #4:   Dihedrals/DIH4.dist.xvg
[(-61, 60)                  # count:   99742 # freq:   0.997 # peak:     3 # mean:     1.116 #
    std:  21.275 # median:     1.647]
```

A list of conformational populations and their relative abundance are provided for the user as a simple text file (*CONFORMATIONAL_POPULATIONS.txt*). Note that a conformer is identified by a ranking number and by a tuple of dihedral populations assigned beforehand and identified here by their respective peaks.

CONFORMATIONAL_POPULATIONS.txt

```
Most common (excluding Z-regions):
P#   1 (27, -89, -10, 3)   : 0.529575 ( 52958)
P#   2 (27, -89, 94, 3)    : 0.382866 ( 38287)
P#   3 (27, -129, 94, 3)   : 0.076269 (  7627)
P#   4 (27, -129, -10, 3)  : 0.000820 (    82)
==
Valid number of frames:      0.989530 ( 98954)
Frames in Z-region:          0.010470 (  1047)
Total number of frames:      1.000000 (100001)
```

5.2.2 Convergence

A folder named **Convergence** contains the sampling evolution as a function of time (*Sampling_Evolution.txt*) and the evolution of relative abundances of each conformer throughout trajectory (*Freq_Conf-*.xvg*). An example of convergence evaluation can be seen in Figure 3.2.

5.2.3 Frames of conformers

A folder named **CONF_Frames** contains a text file with a list of all frames in which a given conformer is observed in MD trajectory (e.g. *Frames_Conf-1.txt*, with frames of the most abundant conformer), as shown below.

Frames_Conf-1.txt

```
(17, −99, 101, 4) # Tuple of dihedral populations relative to this conformer
1660.0
2350.0
2360.0
2650.0
2660.0
2670.0
2680.0
2690.0
2700.0
2710.0
2720.0
3100.0
3180.0
3260.0
```

5.3 Networks

This folder contains information regarding transition events structured as graphs with nodes and edges.

File *CONF_network.txt* contains information regarding node and edges weights (relative abundance and transition frequency, respectively), as shown below.

CONF_network.txt

```
# NODES

Start node: (−138, 0, −143, −66, −68)
End node:   (−138, −174, 133, −66, −68)

(−138, 0, −3, −66, −68)       :   25565.0
(−138, −174, −3, −66, −68)    :   17778.0
(−138, 0, 133, −66, −68)      :    9558.0
(136, 0, −3, −66, −68)        :    8356.0
(70, −174, −3, −66, −68)      :    6644.0
(−138, 0, −143, −66, −68)     :    5712.0
(−138, −174, −143, −66, −68):      3823.0
(−138, −174, 133, −66, −68) :      3147.0
(136, 0, −3, −66, 79)         :    2193.0
(70, 0, −3, −66, −68)         :    2180.0
(136, 0, 133, −66, −68)       :    1923.0
(70, −174, 133, −66, −68)     :    1825.0
(136, 0, −143, −66, −68)      :    1590.0
(136, 0, −143, −66, 79)       :    1513.0
(70, −174, −143, −66, −68)    :    1049.0
(136, −174, −3, −66, 79)      :     965.0
(136, 0, 133, −66, 79)        :     616.0
(70, 0, 133, −66, −68)        :     501.0
...
```

```
# EDGES

((−138, 0, −143, −66, −68), (−138, 0, 133, −66, −68))        :    1428.0
((−138, 0, 133, −66, −68), (−138, 0, −143, −66, −68))        :    1408.0
((−138, 0, 133, −66, −68), (−138, 0, −3, −66, −68))          :    1244.0
((−138, 0, −3, −66, −68), (−138, 0, 133, −66, −68))          :    1229.0
((−138, 0, −3, −66, −68), (−138, 0, −143, −66, −68))         :     898.0
((−138, 0, −143, −66, −68), (−138, 0, −3, −66, −68))         :     883.0
((−138, −174, −143, −66, −68), (−138, −174, 133, −66, −68)):     821.0
((−138, −174, 133, −66, −68), (−138, −174, −143, −66, −68)):     801.0
((136, 0, −3, −66, −68), (70, 0, −3, −66, −68))              :     616.0
((70, 0, −3, −66, −68), (136, 0, −3, −66, −68))              :     612.0
((136, 0, −3, −66, −68), (136, 0, −143, −66, −68))           :     605.0
((136, 0, −143, −66, −68), (136, 0, −3, −66, −68))           :     587.0
((−138, −174, −3, −66, −68), (−138, −174, −143, −66, −68))  :     552.0
((−138, −174, −143, −66, −68), (−138, −174, −3, −66, −68))  :     530.0
((−138, −174, 133, −66, −68), (−138, −174, −3, −66, −68))   :     484.0
((−138, 0, −3, −66, −68), (136, 0, −3, −66, −68))            :     466.0
((−138, −174, −3, −66, −68), (−138, −174, 133, −66, −68))   :     465.0
((136, 0, −3, −66, −68), (−138, 0, −3, −66, −68))            :     458.0
((136, 0, 133, −66, −68), (136, 0, −3, −66, −68))            :     437.0
((136, 0, −3, −66, −68), (136, 0, 133, −66, −68))            :     416.0
((70, −174, −143, −66, −68), (70, −174, 133, −66, −68))     :     377.0
((70, −174, 133, −66, −68), (70, −174, −143, −66, −68))     :     367.0
((70, −174, −3, −66, −68), (136, −174, −3, −66, −68))       :     364.0
((136, −174, −3, −66, −68), (70, −174, −3, −66, −68))       :     354.0
((136, 0, −143, −66, 79), (136, 0, 133, −66, 79))            :     152.0
((136, 0, 133, −66, 79), (136, 0, −143, −66, 79))            :     145.0
((70, 0, 133, −66, −68), (136, 0, 133, −66, −68))            :     118.0
((136, −174, −3, −66, 79), (70, −174, −3, −66, 79))          :     117.0
((70, −174, 133, −66, −68), (136, −174, 133, −66, −68))     :     115.0
((136, −174, 133, −66, −68), (70, −174, 133, −66, −68))     :     114.0
((70, −174, −3, −66, 79), (136, −174, −3, −66, 79))          :     114.0
((136, 0, 133, −66, −68), (70, 0, 133, −66, −68))            :     103.0
...
```

Information regarding nodes and edges are written in *CONF_network.gml*, which is compatible with proper network management softwares, such as Cytoscape [17]. If PLOT_NETWORK in config file is True, *ConfID* roughly draws the transitions network in *CONF_network.eps*, as exemplified in Figure S5.1. The green node represents the initial conformation from MD simulation and the doubled-layered node represents the conformation on the last frame.

Although most *ConfID* users are prone to focus on the conformational transition network mentioned above, the subfolder **Dihedrals** contains information regarding transition events between different dihedral populations, which can be analyzed for each torsion separately. In such cases, file naming follows the same logic described above.

5.4 Time_Dependent_Stats

This folder contains information regarding time-dependent statistics of each conformational population and transition events.

The amount of transition events occurred in each torsion is available in *TRANSITIONS.txt*. File *CONF.transitions.txt* contains the frame in which a conformational transition occurred and the pair of nodes involved, as shown on the scheme below. File *CONF.transitions.xvg* contains a qualitative tracking of transition events as a function of time, in which 1 and 0 represent the presence or absence of a transition event on that frame number.

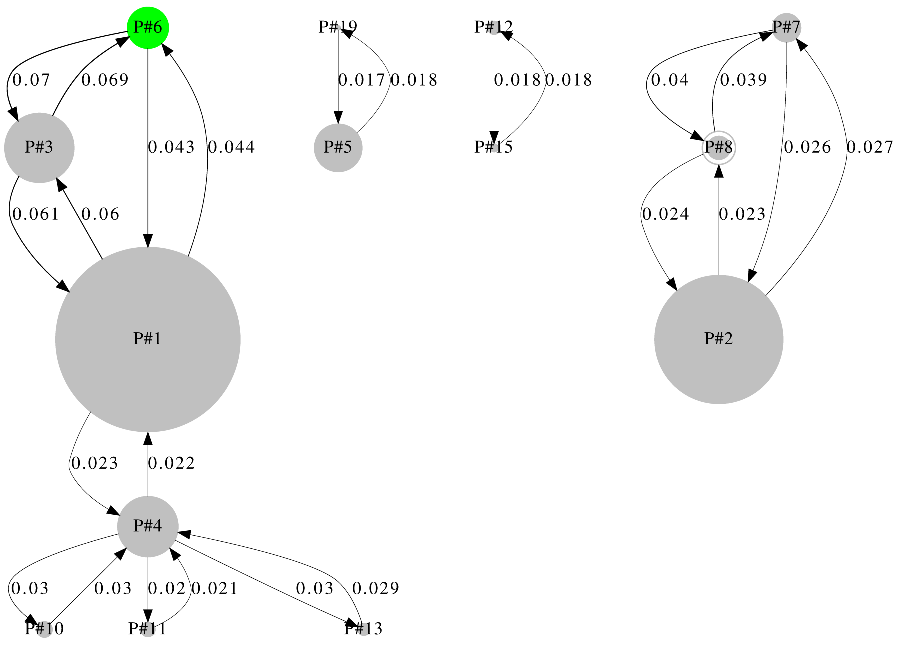


Figure 5.1: Characterization of the conformational dynamics of a glycosilated chalcone simulated in water. Nodes represent conformational populations while edges represent transition events (labels in frequency). Only edges with frequency higher than 1% are shown, along with their connecting nodes. The green node (P#6) is the initial conformation from MD simulation and the doubled-layered node (P#8) is the final one.

CONF.transitions.txt

```
(−96, 76, 58, −174, −176, −71) −> (−96, 76, 58, −174, 135, −71)   | t =      10.0
(−96, 76, 58, −174, 135, −71)  −> (−96, 76, 58, −174, −176, −71)  | t =      15.0
(−96, 76, 58, −174, −176, −71) −> (−96, 76, 58, −174, 135, −71)   | t =      20.0
(−96, 76, 58, −174, 135, −71)  −> (−96, 76, 58, −174, −176, −71)  | t =      25.0
(−96, 76, 58, −174, −176, −71) −> (−96, 76, 58, −174, 135, −71)   | t =      55.0
(−96, 76, 58, −174, 135, −71)  −> (−96, 76, 58, −174, −176, −71)  | t =      60.0
(−96, 76, 58, −174, −176, −71) −> (−96, 76, 58, −174, 135, −71)   | t =      80.0
(−96, 76, 58, −174, 135, −71)  −> (−96, 76, 58, −174, −176, −71)  | t =     130.0
(−96, 76, 58, −174, −176, −71) −> (−96, 76, 58, −174, 135, −71)   | t =     135.0
(−96, 76, 58, −174, 135, −71)  −> (−96, 76, 58, −174, −176, −71)  | t =     170.0
...
```

Time-dependent properties are calculated if TIME_DEPENDENT_STATS in config file is True. File *CONF.transitions-Time_Stats-sumXaver.txt* contains statistics of time-dependent properties of conformers, which are ranked by the property *sum*, as shown on the scheme below. File *CONF.transitions-Time_Stats-sumXaver.xvg* is a graph of properties *sum* (X-axis) and *aver* (Y-axis), which can be visualized in *Xmgrace* and used to calculate correlations. File *CONF.transitions-Time_Stats-sumXaver.png* (Fig. S3.4) is an mere image generated by using matplotlib on *CONF.transitions-Time_Stats-sumXaver.xvg*. Users must bear in mind that any given pair of properties can be used for calculation. To see time-dependent properties available in *ConfID*, see section 4.2.

CONF.transitions-Time_Stats-sumXaver.txt

```
P#   1 (−138, 0, −3, −66, −68)      : # sum: 260010.0   # max: 1080.0   # min:  10.0   # aver:
      95.592    # std: 117.263   # median:   50.000   # count:  2720
P#   2 (−138, −174, −3, −66, −68)   : # sum: 180440.0   # max: 1860.0   # min:  10.0   # aver:
     169.906   # std: 211.434   # median:   90.000   # count:  1062
P#   3 (−138, 0, 133, −66, −68)     : # sum:  96760.0   # max:  340.0   # min:  10.0   # aver:
      34.496    # std:  31.467   # median:   20.000   # count:  2805
P#   4 (136, 0, −3, −66, −68)       : # sum:  84680.0   # max:  350.0   # min:  10.0   # aver:
      37.304    # std:  41.210   # median:   20.000   # count:  2270
P#   5 (70, −174, −3, −66, −68)     : # sum:  67770.0   # max: 1880.0   # min:  10.0   # aver:
     140.894   # std: 219.158   # median:   60.000   # count:   481
P#   6 (−138, 0, −143, −66, −68)    : # sum:  58180.0   # max:  160.0   # min:  10.0   # aver:
      23.757    # std:  19.801   # median:   20.000   # count:  2449
P#   7 (−138, −174, −143, −66, −68): # sum:  39120.0   # max:  180.0   # min:  10.0   # aver:
      28.430    # std:  23.742   # median:   20.000   # count:  1376
P#   8 (−138, −174, 133, −66, −68)  : # sum:  32070.0   # max:  160.0   # min:  10.0   # aver:
      24.631    # std: 20.365    # median:   20.000   # count:  1302
...
```

Moreover, *ConfID* calculates statistics of transition events and time-dependent properties of each dihedral population, allowing the user to analyze each torsion separately. These outputs can be found in the subfolder **Dihedrals**. In such cases, file naming follows the same logic described above. Note that the previous pair of selected properties are used to calculate time-dependent statistics for the dihedral population as well.

CHAPTER 6

Installation

6.1 Install *ConfID* on Ubuntu via snap (recommended)

6.1.1 Enable snapd

If you are using Ubuntu 16.04 LTS (Xenial Xerus) or later, including Ubuntu 18.04 LTS (Bionic Beaver), Ubuntu 18.10 (Cosmic Cuttlefish) and Ubuntu 19.04 (Disco Dingo), Snap should be already installed.

For versions of Ubuntu between 14.04 LTS (Trusty Tahr) and 15.10 (Wily Werewolf), as well as Ubuntu flavors that do not include snap by default, it can be installed from the Ubuntu Software Centre by searching for snapd, or from the command line:

```
$ sudo apt update
$ sudo apt install snapd
```

Either log out and back in again or restart your system, to ensure snap's paths are updated correctly.

6.1.2 Install *ConfID*

To install *ConfID*, use the following commands on your terminal:

```
$ sudo snap install confid
$ snap connect confid:removable-media
```

Confirm *ConfID* is installed by listing your installed snaps:

```
$ snap list
```

Then you can run *ConfID* by typing on your terminal:

```
$ confid input.inp config
```

6.2 Install *ConfID* on other Linux distributions via snap

To install *ConfID* in other Linux distributions (Fedora, Linux Mint, Kubuntu, Debian, etc.), please refer to the instructions at the bottom of the page: https://snapcraft.io/confid.

6.3 Install *ConfID* on macOS using the exec file (recommended)

You can run *ConfID* on your macOS machine by using the provided exec file.

6.3.1 Download the exec

Download the file confid_1.2.0_macos to your directory of choice: https://github.com/sbcblab/confid/blob/master/confid_1.2.0_macos
Rename it to "confid" for practicality.

6.3.2 Run *ConfID*

To run, type on your terminal in the same directory of the downloaded exec:

```
./confid -h
```

6.4 Install *ConfID* on macOS using Snapcraft

Snapcraft 3 can be installed via Homebrew on Apple macOS Yosemite (or later), and then used to build *ConfID* within the macOS environment.

6.4.1 Prerequisites

Install the the Homebrew package manager: https://brew.sh/#install

Install Snapcraft and Multipass by opening 'Terminal' and enter the following:

```
$ brew install snapcraft
$ brew cask install multipass
```

6.4.2 Configure the virtual machine

Create a Linux virtual machine:

```
$ multipass launch -n testvm
```

Connect to the virtual machine:

```
$ multipass shell testvm
```

6.4.3 Install the snap

Install *ConfID* inside the virtual machine:

```
$ sudo snap install confid
$ snap connect confid:removable−media
```

Confirm *ConfID* is installed by listing your installed snaps:

```
$ snap list
```

Run *ConfID* inside the virtual machine:

```
$ confid −h
```

Exit the virtual machine:

```
$ exit
```

6.5 Install *ConfID* from the source code

This is the best way to use *ConfID* as a developer if you wish to modify the source code yourself, or if you want to use *ConfID* on other operating systems.

6.5.1 Check for dependencies

To run *ConfID*, you will need either Python 2.7 or Python 3.x installed in your machine. *ConfID* requires the following external libraries:

- graphviz (https://pypi.org/project/graphviz/)
- matplotlib (https://matplotlib.org/)
- numpy (https://numpy.org/)

If any of those are missing, you should install them before continuing. Before installing Graphviz for Python, make sure that the original software is installed (http://www.graphviz.org/download/). Before installing it, the following dependencies should be installed as well:

```
$ sudo apt install −y libgd−dev
$ sudo apt install −y fontconfig
$ sudo apt install −y libcairo2−dev
$ sudo apt install −y libpango1.0−dev
$ sudo apt install −y libgts−dev
```

6.5.2 Download *ConfID*

Download all files from https://github.com/sbcblab/confid and save them to your directory of choice. If you have git installed you can do this by typing the following in your terminal:

```
$ git clone https://github.com/sbcblab/confid.git
```

Now you can run *ConfID* by typing the following command in your terminal:

```
$ python3 confID.py input.inp config
```

6.5.3 Aliasing *ConfID* (optional)

To run *ConfID* in any given directory without carrying many files with you, we strongly advise you to alias *ConfID*. For this:
Copy the "confid" folder that was extracted before;
Move it to an installation folder of your choice;
Then, add the following line to the end of your /.bashrc file:

```
alias confid='python3 /installation\_folder\_you\_chose/ConfID
    /confID.py'

(example: alias confid='python3 /home/user/Tools/ConfID/confID
    .py)
```

Then run on terminal:

```
$ source ~/.bashrc
```

After this, you should be able to run *ConfID* in your terminal at any given directory by simply typing:

```
$ confid input.inp config
```

CHAPTER 7

Running *ConfID*

To run *ConfID* open your terminal and type

7.0.1 From Snapcraft

```
$ confid input.inp config
```

7.0.2 From macOS exec

```
$ ./confid input.inp config
```

7.0.3 From source (not aliased)

```
$ python3 confID.py input.inp config
```

7.0.4 From source (aliased)

```
$ confid input.inp config
```

In which "input.inp" is a file containing in each line the paths to each dihedral file to serve as input, and "config" is an optional file containing the configuration of parameters for *ConfID.*

To read the help section of *ConfID* you can also run

```
$ confid -h
```

CHAPTER 8

Tutorials

In order to take full advantage of *ConfID* features and capability, we provide different tutorials for the users. You can access the tutorials here: `https://github.com/sbcblab/confid/blob/master/TUTORIAL.md`

CHAPTER 9

Solution to errors and warning messages

Assuming that all prerequisites were correctly installed (check the Installation section) and you have tried the tutorials, the most common errors and warnings are:

1. *"Segmentation Fault"*

   In general, that means *ConfID* is using too much memory, most probably due to the number of frames saved in your inputs. You might want to try to close some applications or to reduce the frequency of frames in your inputs.

2. *"ERROR: wrong number of parameters given to ConfID: To run ConfID, please provide the path to the input.inp and optionally the path to the config file."*

   This error means that the wrong number of inputs was given to *ConfID*. *ConfID* accepts the following commands:

   ```
   $ confid
   $ confid -h
   $ confid input.inp
   $ confid input.inp config
   ```

3. *"PermissionError: IMPORTANT! Are you trying to read or write files in a removable media (pendrives, external HDs, etc)? Then you must give ConfID access to removable media after installing it by running the following command in your terminal: snap connect confid:removable-media"*

   This error happens if you install *ConfID* using the snapcraft, try to read or write a file from a removable media, but did not give *ConfID* access to removable media. To solve it type on your terminal:

   ```
   $ snap connect confid:removable-media
   ```

4. *"ERROR: wrong number of columns in the dihedral file!* ConfID only reads files with one (dihedral value) or two columns (time and dihedral value)."

This means that one or more of your files with dihedral values have more than two columns in at least one line. Please check your files and remember that ConfID accepts dihedral files with one (dihedral value) or two columns (time and dihedral value).

5. "*ERROR: extra value in the dihedral file!* ConfID only reads files with one (dihedral value) or two columns (time and dihedral value)."

   This means that you are using dihedral files with only one column (dihedral value) but at least one line has more than one value.

6. "*ERROR: missing value in the dihedral file!* ConfID only reads files with one (dihedral value) or two columns (time and dihedral value)."

   This means that you are using dihedral files with two columns (time and dihedral value), but at least one line has less than two values.

7. *"ERROR: conflicting number of timesteps in the dihedral file."*

   This means that at least one dihedral file from your input.inp has a different number of timesteps. Please ensure that all your input files have the exact same number of simulation timesteps.

8. *"ERROR: Wrong number of columns in file: Config file must have only two columns separated by space."*

   This means that your config file has a line with less than or more than two columns. The optional config file requires exactly two columns separated by spaces.

9. *"ERROR: graphviz package needs to be installed if PLOT_NETWORK is True, but it couldn't be imported."*

   This will happen if you try to plot networks within *ConfID* but does not have the Graphviz package installed in your system. You can either install Graphviz manually (please refer to `https://graphviz.readthedocs.io/en/stable/manual.html`) or set PLOT_NETWORK to False and use the .gml files generated in a network visualization software such as Cytoscape [17].

10. *"ERROR: matplotlib.pyplot package needs to be installed if TIME_DEPENDENT_STATS is True, but it couldn't be imported."*

    This will happen if you don't have the matplotlib package installed in your system. You can solve this by installing matplotlib (`https://matplotlib.org/3.1.0/users/installing.html`) or setting TIME_DEPENDENT_STATS to False.

11. *"ERROR: SIM_TIME must be 'None' or a value larger than 0.0!"*

    To solve this problem, set the SIM_TIME value in your config file to "None" or to be larger than 0.0.

12. *"ERROR: WINDOW_LEN must be odd integer between 3 and 180."*

    To solve this problem, set the WINDOW_LEN value in your config file to an odd integer equal or larger than 3 and smaller than 180.

13. *"ERROR: WINDOW must be one of 'flat', 'hanning', 'hamming', 'bartlett', 'blackman'"*

    To solve this problem, set the WINDOW value in your config file to one of the strings above, without the quotation marks.

14. *"ERROR: FACTOR_PEAK must be larger or equal to 1.0."*

    To solve this problem, set the FACTOR_PEAK value in your config file to be larger or equal to 1.0.

15. *"ERROR: FACTOR_VALLEY must be larger or equal to 1.0."*

    To solve this problem, set the FACTOR_VALLEY value in your config file to be larger or equal to 1.0.

16. *"ERROR: FACTOR_VALLEY must be larger than FACTOR_PEAK"*

    To solve this problem, set the FACTOR_VALLEY value in your config file to be larger than FACTOR_PEAK.

17. *"ERROR: No peaks were found! Try using a larger threshold value for FACTOR_PEAK to solve the problem."*

    This error will happen if you use a FACTOR_PEAK so small that no peaks could be found.

18. *"ERROR: No valleys were found! Try using a larger threshold value for FACTOR_VALLEY to solve the problem."*

    This error will happen if you use a FACTOR_VALLEY so large that no peaks could be found.

19. *"ERROR: total simulation time in the dihedral file is different from simulation time in the config file!"*

    There is a value other than "None" for SIM_TIME in the config file, and the dihedral files have two columns, one of them being the time steps. The values in these files are different, so one of them must be wrong.

20. *"ERROR: total simulation time was not declared in the config file, but the dihedral file has no time column!"*

    The SIM_TIME was equal to "None" or was not specified in the config file, but the dihedral files do not have a time column. Please set SIM_TIME to the total simulation time.

21. *"ERROR: Unidentified function:"*

    This error will happen when an unidentified function is passed as an argument to DATA_1 or DATA_2 in the config file. The possible values are sum, max, min, aver, std, median, and count.

22. *"ValueError: Input vector needs to be bigger than window size."*

    This error will happen only if the file containing the dihedral distributions (*.dist.xvg) has less than 21 (or the value of WINDOW_LEN) angles listed.

23. *"WARNING: Unidentified parameter ignored"*

    This means that some parameters in your config file could not be recognized and were ignored. Please review the spelling of the parameters and check if they match the documentation.

24. *"WARNING: declared simulation time in the config file and in the dihedral file at the same time."*

    This is a reminder that there is a value other than "None" for SIM_TIME in the config file, and the dihedral files have two columns, one of them being the time steps. If the values in these files do not match up, an error will arise.

25. *"WARNING: plotting the graph figures may become too slow if the Z populations are to be shown, please consider setting either SHOW_Z or PLOT_NETWORK to False or to use a large NETWORK_CUTOFF."*

    This warning appears if both SHOW_Z and PLOT_NETWORK are True.

26. *"WARNING: the frequency of the conformational populations in the stay stats may vary slightly from the previous results if Z populations are disregarded."*

27. *"WARNING: Did not plot network with more than 200 nodes."*

    This warning appears to avoid memory and processing time issues by plotting a huge network. It occurs when PLOT_NETWORK is True, and the network has more than 200 nodes. You can use the .gml files generated in a network visualization software such as Cytoscape [17].

    If you are having trouble installing or running *ConfID*, we kindly ask you to send your inputs and a brief description of what you are trying to achieve (organized screenshots may help) to marcelo.poleto@ufv.br and bigrisci@inf.ufrgs.br and our team will deal with it as soon as possible.

Bibliography

[1] Sereina Riniker and Gregory A. Landrum. Better Informed Distance Geometry: Using What We Know To Improve Conformation Generation. *Journal of Chemical Information and Modeling*, 55(12):2562–2574, dec 2015.

[2] Adriana Supady, Volker Blum, and Carsten Baldauf. First-Principles Molecular Structure Search with a Genetic Algorithm. *Journal of Chemical Information and Modeling*, 55(11):2338–2348, nov 2015.

[3] Charles H. Reynolds. Protein–Ligand Cocrystal Structures: We Can Do Better. *ACS Medicinal Chemistry Letters*, 5(7):727–729, jul 2014.

[4] Andria L. Skinner and Jennifer S. Laurence. High-field solution NMR spectroscopy as a tool for assessing protein interactions with small molecule ligands. *Journal of Pharmaceutical Sciences*, 97(11):4670–4695, nov 2008.

[5] M.D. Polêto, V.H. Rusu, B.I. Grisci, M. Dorn, R.D. Lins, and H. Verli. Aromatic Rings Commonly Used in Medicinal Chemistry: Force Fields Comparison and Interactions with Water Toward the Design of New Chemical Entities. *Frontiers in Pharmacology*, 9(395):395, 2018.

[6] Pablo R. Arantes, Marcelo D. Polêto, Elisa B. O. John, Conrado Pedebos, Bruno I. Grisci, Marcio Dorn, and Hugo Verli. Development of Gromos-Compatible Parameter Set for Simulations of Chalcones and Flavonoids. *The Journal of Physical Chemistry B*, 123(5):994–1008, feb 2019.

[7] Jožica Dolenc, Wilfred F. van Gunsteren, Andrea E. Prota, Michel O. Steinmetz, and John H. Missimer. Conformational Properties of the Chemotherapeutic Drug Analogue Epothilone A: How to Model a Flexible Protein Ligand Using Scarcely Available Experimental Data. *Journal of Chemical Information and Modeling*, 59(5):2218–2230, may 2019.

[8] Jihyun Shim and Alexander D. MacKerell, Jr. Computational ligand-based rational design: role of conformational sampling and force fields in model development. *MedChemComm*, 2(5):356, 2011.

[9] Satoshi Ono, Matthew R. Naylor, Chad E. Townsend, Chieko Okumura, Okimasa Okada, and R. Scott Lokey. Conformation and Permeability: Cyclic Hexapeptide Diastereomers. *Journal of Chemical Information and Modeling*, 59(6):2952–2963, jun 2019.

[10] Sereina Riniker, Clara D. Christ, Halvor S. Hansen, Philippe H. H??nenberger, Chris Oostenbrink, Denise Steiner, and Wilfred F. Van Gunsteren. Calculation of Relative Free Energies for Ligand-Protein Binding, Solvation, and Conformational Transitions Using the Gromos Software. *Journal of Physical Chemistry B*, 115(46):13570–13577, 2011.

[11] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. GROMACS: High Performance Molecular Simulations Through Multi-level Parallelism from Laptops to Supercomputers. *SoftwareX*, 1:19–25, 2015.

[12] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD: Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14(1):33–38, feb 1996.

[13] James C. Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D. Skeel, Laxmikant Kalé, and Klaus Schulten. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 26(16):1781–1802, dec 2005.

[14] F. J. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51–83, Jan 1978.

[15] Paul C. D. Hawkins, A. Geoffrey Skillman, Gregory L. Warren, Benjamin A. Ellingson, and Matthew T. Stahl. Conformer Generation with OMEGA: Algorithm and Validation Using High Quality Structures from the Protein Databank and Cambridge Structural Database. *Journal of Chemical Information and Modeling*, 50(4):572–584, apr 2010.

[16] Paul C. D. Hawkins. Conformation Generation: The State of the Art. *Journal of Chemical Information and Modeling*, 57(8):1747–1756, aug 2017.

[17] P. Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Research*, 13(11):2498–2504, nov 2003.