In this tutorial, we are going to analyze conformations of radicicol (RDC), a macrocyclic non-covalent inhibitor binding to the Heat shock protein 90 (Hsp90). The dynamics of RDC were simulated in explicit water with molecular dynamics (MD) to generate different conformations. The ClassTor approach was designed and thoroughly tested for the classification of conformations of macrocycles generated with MD. However, the application is not limited to this setup, see comments on the performance of ClassTor below.

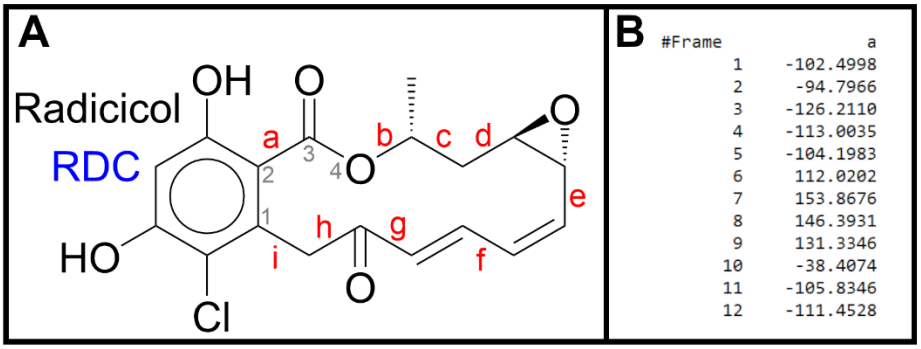

Figure 1. A) radicicol, torsions considered for classification labeled with red letters. Atom numbers in grey **B)** excerpt of the dihedral angle file *a_angles.dat*.

For this tutorial, the required torsion angle files are provided in the *example/* directory of the repository. In your own case, the following preparation steps are necessary before using ClassTor:

1. Select a set of torsions considered for classification and label them with consecutive letters (**Figure 1A**).

   These should include potentially rotatable (ideally: flexible) torsions. Therefore, ester (and amide) bonds, double bonds, and bonds that are part of other ring systems can be excluded for their limited flexibility.
2. Extract the dihedral angle values of such torsions and save them in separate two-column files including a single header line (for example with cpptraj or vmd). These files must be named according to the label of their torsion: *a_angles.dat*, *b_angles.dat*, *c_angles.dat* and so on. An exemplary excerpt of *a_angles.dat* is shown in **Figure 1B** generated with the following cpptraj command:

   ```
   dihedral a @1 @2 @3 @4 out a_angles.dat
   ```

3. If desired, add a reference torsion angle value as the last line at the bottom of each file. For example, the corresponding values of the bound ligand conformation. If so, *-x* must be enabled on the command line when executing *ClassTor.py* to exclude this frame for the generation and characterization of the torsion angle distribution spectra and classification. The reference is

only considered for comparison when performing distance calculations between all sampled frames.

For this tutorial, the original MD trajectory of RDC of 30,000 frames was sieved to 1,000 frames and the torsion angle values of labeled torsions were extracted with cpptraj as described above. The corresponding value of the bound conformation was added at the end of each file as a reference. Download the files from the repository and store them together with the *ClassTor.py* analysis script in a directory of your choice and change to that directory.

Note: *ClassTor.py* imports the two module files *classtor_spectrum.py* and *classtor_classification.py*. These files must be stored in your python path; see "**Installation**" in the *README.md* file in the ClassTor repository on GitHub.

Execute the following command:

```
./ClassTor.py 9 -gk 30 -t 30 -x
```

The number of torsions is 9 (*a-i* in **Figure 1A**, *AngNum* see documentation). ClassTor automatically converts this number to a range of 9 consecutive letters starting from "*a*" and reads in the corresponding torsion angle files. In this case, *-gk* 30 and *-t* 30 generate a smooth spectrum with meaningful bin definitions. As mentioned above, *-x* interprets the last line of every file as the reference value for that torsion. Default values for other optional flags were employed (*-pa* ./ *-ss* 10 *-sr* 0 *-so* random) and the following output files were generated:

1. A distribution spectrum for each torsion: *spectrum_a.dat – spectrum_i.dat*
2. A summary of the characterization of all spectra and the flexibility scoring: *summary_spectrum_gk30_t30.log*
3. The classification output: *summary_classification_gk30_t30.log*
4. Two files for the heatmap comparison between structures of the conformer subsets: *RMS2TA_10class_10diverse.gnu*, *heatmap_10class_10diverse.in*

These files can be found in the *results_gk30_t30/* directory on the repository for comparison. For demonstration, we repeated the procedure with the default settings for *-gk* (15) and *-t* (20):

```
./ClassTor.py 9 -x
```

The results of which can be found in *results_gk15_t20/* on the repository. After every execution, carefully inspect if the generated spectra match their characterization. Therefore, plot the spectrum (e.g., with xmgrace) and compare its course with the information saved in the *summary_spectrum.log* files. Exemplary shown for torsions *a* and *d*, respectively, the optimized parameters (*-gk* 30 *-t* 30) generated a smooth spectrum (**Figures 2A** and **3A**, respectively), where the definition of bins is more

meaningful than for the default values (**Figures 2B** and **3B**, respectively). A more detailed illustration is given in the following:

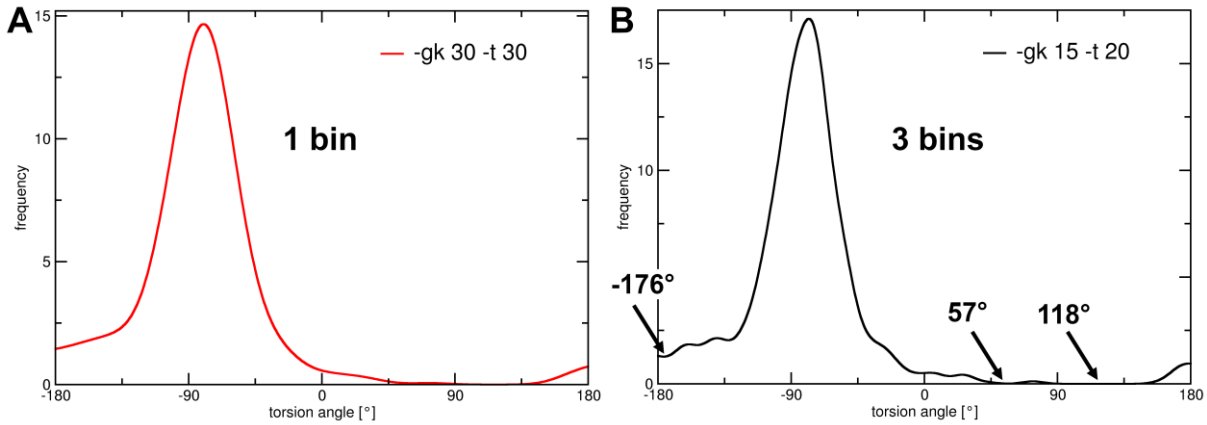

Figure 2. Distribution spectrum of dihedral angles of torsion *d* (see **Figure 1A**). **A)** -gk 30 -t 30. **B)** ClassTor default settings.

In the case of torsion *d*, the spectrum is divided in 1 bin (3 bins) for the optimized (for the default) settings, see **Figure 2**. The default settings detected small fluctuations as relative extrema (minima at -176°, 57° and 118°) resulting in a less meaningful division as torsion angle values of very low frequency are divided into separate bins. More intuitively, the spectrum is divided in 1 bin with a smoother path generated by the optimized settings.

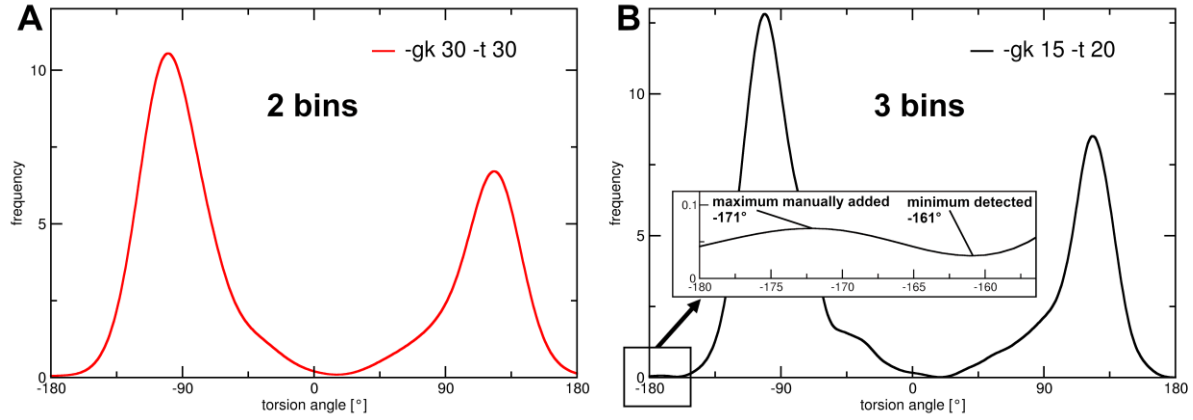

Figure 3. Distribution spectrum of dihedral angles of torsion *a* (see **Figure 1A**). **A)** -gk 30 -t 30. **B)** ClassTor default settings.

Furthermore, torsion *a* was divided into 3 bins with the default settings, where bin 0 is defined by a very small range at -180° (**Figure 3B**). Additionally, the *information* column in *summary_spectrum_gk15_t20.log* says that this bin was missing a center value, and that a maximum at -171° was added manually. The maximum was not detected by *argrelextrema* as the highest frequency value within a range of 20° (20 neighboring points defined via -*t* 20) due to the rapid increase

in frequency of the neighboring peak. For distance calculations between torsion angle values and bin centers, every bin must have exactly one maximum value. For flat courses like in bin 0, such a correction is acceptable and was therefore implemented in Classtor. However, if this is the case for more than one bin of a torsion, the program stops, assuming that the shape of the distribution spectrum prevents a useful division. Thus, the comments in the *information* column must be examined carefully and the torsion angle spectra should be visually inspected for such corrections and re-shaped with different values for *-gk* and *-t* if necessary.

Testing different combinations is highly recommended to find the most meaningful characterization and accepting the classification results without inspecting the generated distribution spectra should be avoided. If unsure which characterization is suitable, carefully check your trajectory and the corresponding torsion angle values to assess if these nuances of the spectrum are important. Here, the bin definitions for torsions *a* and *d* are too detailed since areas with low frequency values (see y-axis) were divided into separate bins. In this case, these fluctuations are artifacts due to the number of frames considered (1,000 sieved frames). This is shown in **Figure 4**, where we repeated the ClassTor procedure with default settings considering the original 30,000 frames. There, these nuances cleared out and were not divided into separate bins, highlighting the importance of a smooth spectrum by considering statistically significant number of structures for easy and robust analysis. As the spectra and the corresponding bin definitions generated with the optimized settings resemble those generated with 30,000 frames, the optimized settings are confirmed to be more suitable in the case of 1,000 frames.

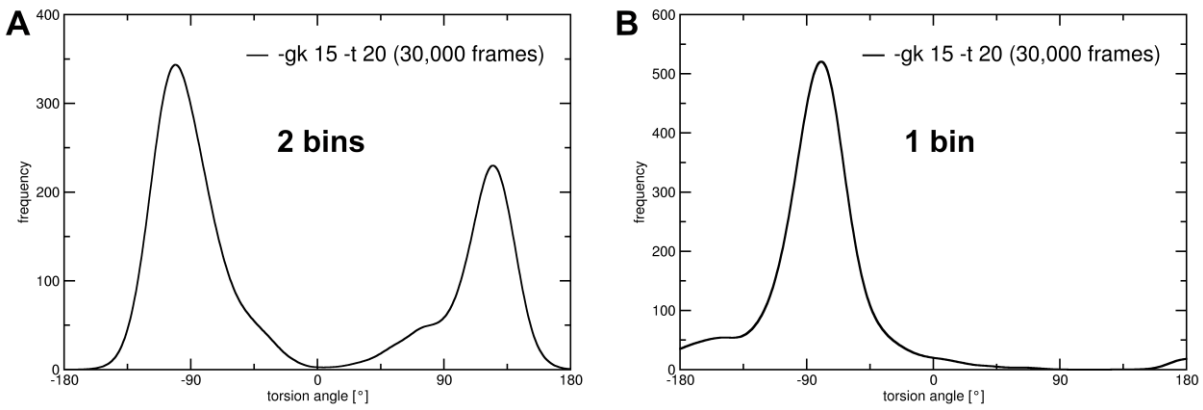

Figure 4. Distribution spectrum of dihedral angles of torsion *a* (**A**) and *d* (**B**) generated with default settings and 30,000 frames.

From the output file *summary_spectrum_gk30_t30.log* we can see that the whole range of dihedral angle values for torsion *d* was only divided into one bin. Such torsions don't have an influence on the classification, as the partitioning into classes is driven by the combination of different bin labels for different torsions. With only one bin label there are no other combinations possible, as shown in the

classifier column of *summary_classification_gk30_t30.log*. Thus, a torsion divided into one bin can be excluded from classification.

The file *summary_classification_gk30_t30.log* is sorted according to the population (*size*) of each class in frames, listing the *classifier* of that class (each frame of a class is characterized by this combination of bin labels for all torsions), a centroid (*centr*) as the frame with the minimum rms-distance of all dihedral angle values of all torsions to the corresponding midpoints of the bin (frames are counted from 1) and a rounded fraction of the number of frames in percent. In our case, the 1,000 frames were classified in 38 classes. Thus, we could reduce the amount of data, e.g., by considering only the 38 centroid frames without losing information about the conformational space. Depending on the number of torsions considered for classification, the number of bins for each torsion and the sampled conformational space, the classification results can be very extensive producing hundreds of classes. Thus, there are certain ways of further reducing the classification output. A common way is to only consider the most dominant conformations (the centroids of the highest populated classes). We also implemented a perturbational approach in ClassTor (see documentation) that extracts a set of diverse conformers by using the structural information decoded in the classifier. Per default (*-ss* 10 *-so* random *-sr* 0), a subset of 10 structures is generated, the class centroids of the highest populated classes and those chosen for the diverse subset. The frame IDs of the structures in each subset are printed to the standard output and saved in the header of *RMS2TA_10class_10diverse.gnu*. The latter is a pairwise rms-distance matrix (in dihedral angle space) between the structures in those subsets that can be plotted with the corresponding gnuplot input file *heatmap_10class_10diverse.in* for a visual representation of this quantitative comparison.

Inspect the heatmap file. Three sections are important to examine before plotting: LABELS, TICS, and PLOT. Per default, the axis LABLES are set to ‘frame ID’ (set xlabel, set ylabel) and the TICS are the corresponding frame IDs of the centroid structures of the 10 highest populated classes (set xtics, set ytics). This subset is plotted in the PLOT section (“using 1:2:3”) with the corresponding matrix file (*RMS2TA_10class_10diverse.gnu*). These settings mean that columns 1, 2 and 3 are used for generating the heatmap plot which corresponds to the subset of the centroids of highest populated classes. If the diverse subset should be plotted, change the *splot* command to “using 1:2:4”, see explanation in the comments. But make sure to change the TICS to the corresponding “frame ID of 10 diverse class centroids”. Alterantively, for each subset, the class IDs can be used as tics instead of the original frame ID. Be careful to always choose the correct tics to the subset you are plotting.

```
gnuplot -p < heatmap_10class_10diverse.in
```

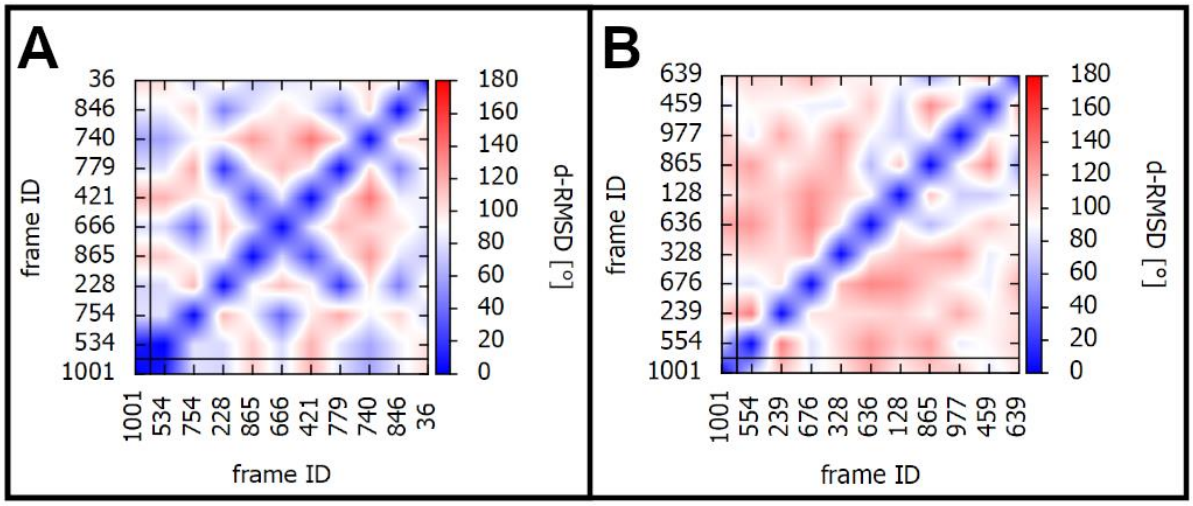

Figure 5. Heatmap comparison (pairwise rms-distance matrix between structures in a conformer subset) by ClassTor, reference (frame 1001) separated for clarity. **A)** Subset consists of the centroid structures of the 10 highest populated classes. **B)** Diverse subset generated with ClassTor's perturbational approach.

Optionally, all class centroid structures obtained from classification can be clustered in a hierarchical manner to produce an alternative subset. This option is enabled via *-c* on the command line. The resulting clusters and their centroids are marked in the *summary_classification_gk30_t30_c.log* output file. Correspondingly, the subset is incorporated in the matrix file *RMS2TA_10class_10clus_10diverse.gnu* as 4th column (the diverse subset is moved to the 5th column). Note, that if you re-run the classification with the same settings for the perturbational approach (*-so* random) the resulting diverse subset will be different than the one from the first run. With a random order of classifiers employed by the approach, the subsets are not reproducible. However, the degree of diversity in each new diverse subset should be similar, see documentation. Thus, if you want to keep the first diverse subset, you might want to create different directories for each run and specify the location of the torsion angle files via *-pa* or copy them. We performed the following command in a separate folder:

```
./ClassTor.py 9 -gk 30 -t 30 -x -c
```

And the corresponding output files are provided in *results_gk30_t30_c/* on the repository. All 3 subsets generated in that way can be plotted with the same matrix (.gnu) and gnuplot input file (heatmap.in) and are shown in **Figure 6**. Note, that some of the structures in the diverse subset (**Figure 6C**) are even similar but can consist of different structures than in **Figure 5B** with a similar degree of conformational diversity. It is highly recommended to try different settings until finding the appropriate subset. If the focus is on the most dominant structures, the centroids of the highest populated classes are suitable (**Figure 6A**), whereas the most diverse subset can be extracted with the perturbational approach

(**Figure 6C**). The subset extracted with post-classification clustering (**Figure 6B**) would be somewhere in between and might be more suitable for classification results with large number of classes.

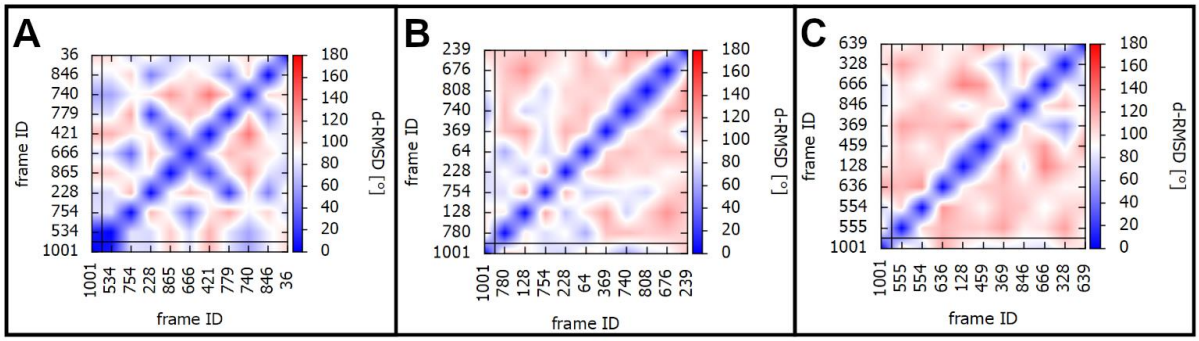

Figure 6. Same as **Figure 5. A)** Subset consists of the centroid structures of the 10 highest populated classes. **B)** Subset of 10 cluster centroids after clustering all classification centroids. **C)** Diverse subset generated with ClassTor's perturbational approach.

Finally, another way to change the classification, and thus the generated subsets, is to only consider a subset of torsions, limiting the possible combinations of bin labels and resulting in fewer classes. Therefore, the flexibility scoring of *summary_spectrum_gk30_t30.log* can be used to decide which torsions to focus on for classification. This approach is recommended if you want to look at many torsions of your molecule first but focus only on the most important for the classification. These torsions can be specified via *-f* on the command line followed by a string of labels (letters) separated by whitespace of the torsions you want to consider for classification. Use the following command:

```
./ClassTor.py 9 -gk 30 -t 30 -x -f a f g h
```

Note, that for the calculation of the silhouette coefficient as well as for distance calculations for comparing the structures in the subset, the dihedral angle values of all torsions (specified via *AngNum*, here: 9) will be used, independent of the *-f* option. The respective output files are provided in the *results_gk30_t30_afgh/* directory on the repository. Naturally, the number of classes reduced, since only 4 torsions were considered for classification. Thus, a new partition is created with new centroids and subsets.