MD DaVis

Release 0.3.0

Dibyajyoti Maity

Oct 02, 2021

CONTENTS

MD DaVis is a tool and Python 3 package to easily create helpful interactive visualizations to analyze and compare multiple molecular dynamics simulations of proteins.

CHAPTER

ONE

DOCUMENTATION

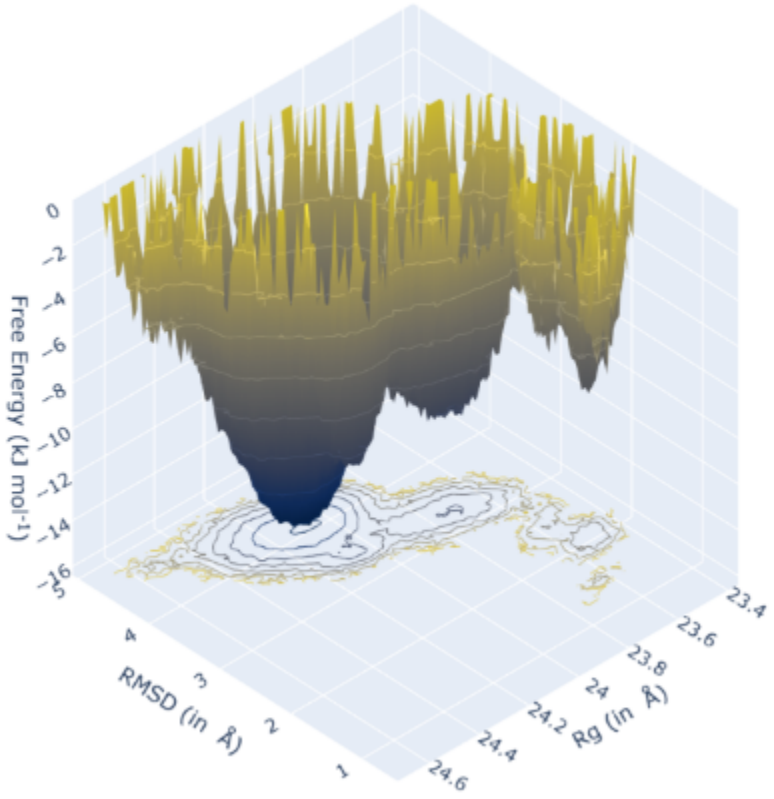
1.1 Introduction

Molecular dynamics (MD) simulations are indispensable for gaining atomistic insights into the biomolecular function. MD of increasingly larger proteins has become accessible to researchers with recent advancements in computing and MD algorithms. However, the analysis of MD trajectories still remains tedious. MD DaVis (Molecular Dynamics Data Visualizer) is a tool and Python 3 package to perform comparative data analysis of MD trajectories of similar proteins or the same protein under different conditions.

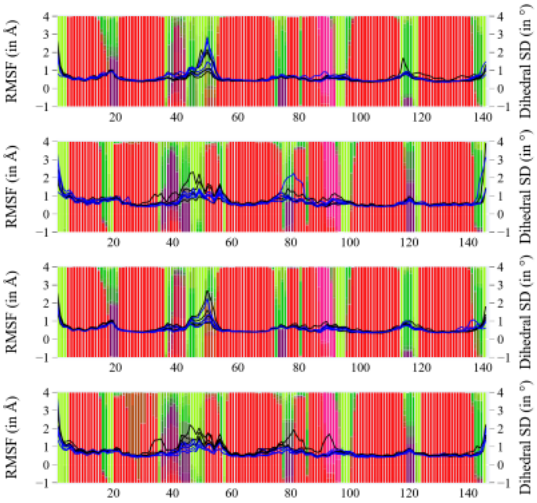
There are many *MD analysis tools*. However, the output from most has to be visualized using another plotting library requiring a significant amount of coding. The MD DaVis package provides a command-line tool to create helpful interactive visualizations easily. The tool can increase the productivity of researchers using MD simulations and make the analysis of such simulations accessible to everyone.

1.1.1 Features of MD DaVis

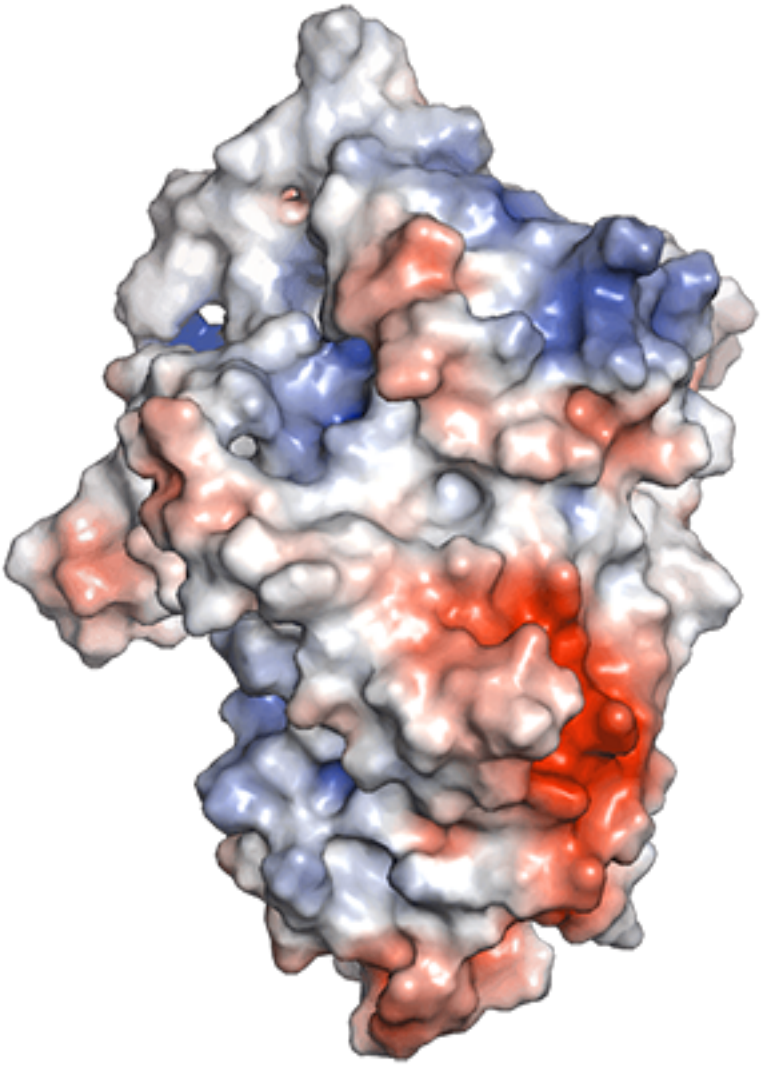
Free Energy Landscapes



Residue Properties Plot

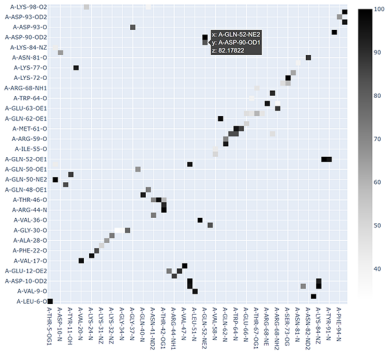


Surface Electrostatics



Electric Field Dynamics

H-bond / Contact Matrix



1.1.2 Molecular Dynamics Analysis Tools

MD DaVis does not implement analysis tools and functions available elsewhere. Presently, MD DaVis natively supports analysis performed using GROMACS tools. However, other analysis tools may be used as well, depending on the format of the MD trajectory. In such cases, the output may be formatted to that accepted by MD DaVis. The following is a non-exhaustive list of tools for analysis of MD simulations:

- MDTraj (McGibbon et al., 2015)
- MDAnalysis (Michaud-Agrawal et al., 2011)
- MD-TASK (Brown et al., 2017)
- TRAVIS (Brehm and Kirch-ner, 2011)
- MDTRA (Popov et al., 2013)
- Bio3d (Grant et al., 2006)
- MDplot (Margreitter and Oostenbrink, 2017)
- GROMACS (Berendsen et al., 1995)
- AMBER (Case et al., 2020)

1.2 User Guide

1.2.1 Quickstart

Getting Help

help for each command in MD Davis can be accessed with `--help` or `-h`, for example:

```
md_davis -h
```

Usage

To use MD DaVis in a project

```
import md_davis
```

Step 1: Calculate the required quantities using GROMACS

Root mean squared deviation (RMSD) and radius of gyration are not required for creating overlaid residue plot, but they would be used to create energy landscapes. Ensure that the processed trajectory and structures are provided, in which the periodic boundary have been corrected for.

Root mean squared deviation (RMSD)

The structure provided here will be used as the reference for calculating RMSD of the trajectory.

```
gmx rms -f trajectory.trr -s structure.gro -o rmsd.xvg
```

Radius of gyration (R_G)

```
gmx gyrate -f trajectory.trr -s structure.gro -o rg.xvg
```

Root mean squared fluctuation (RMSF)

```
gmx rmsf -f trajectory.trr -s structure.gro -res -o rmsf.xvg
```

This would align the whole molecule for RMSF calculation. You may calculate the RMSF of each chain separately in a protein with multiple chains. This would reflect the fluctations only due to backbone motion and elimiate the fluctuation from relative motion of chains.

Solvent accessible surface area (SASA)

To calculate the average SASA per residue pass `-or` option to `gmx sasa`.

```
gmx sasa -f trajectory.trr -s structure.gro -o sasa.xvg -or resarea.xvg
```

Secondary structure using DSSP

For the GROMACS command `do_dssp` to work the DSSP binary must be available on your system. [Download DSSP binaries here](ftp://ftp.cmbi.ru.nl/pub/software/dssp/)

```
gmx do_dssp -f trajectory.trr -s structure.gro \
        -o dssp \
        -ssdump dssp \
        -sc dssp_count
```

`-ssdump` option will output a file `dssp.dat` containing the secondary structure for the full trajectory as single letter codes. This would be processed by MD DaVis to calculate the secondary structure persistence at each residue.

Code	Secondary structure
H G I B E T S ~	-helix 3₁₀-helix -helix -bridge strand Turn Bend Loop

Step 2: Collate into HDF5 file

Step 2a: Obtain the sequence of the protein from the PDB file used to start the simulation.

```
md_davis sequence structure_used_for_simulation.pdb
```

Step 2b: Provide this sequence in JSON file below, along with a few other properties. Note that for multi-chain proteins the sequence for each chain would be separated by a '/'.

```
{
    "label": "MD Simulation",
    "short_label": "MD",
    "html": "<i>MD Simulation</i>",
    "short_html": "<i>MD Simulation</i>",
    "protein": "protein name",
    "scientific_name": "some organism",
    "common_name": "common name",
    "sequence": "PUT/YOUR/SEQUENCE/HERE"
}
```

The most important property here is the **sequence**, which tells `md_davis collect` of the number of chains in the molecule and the number of residues in each chain. The **short_html** will determine the labels for the data in the final plots. This file is named `information.json` in the next command.

Step 2c: Collect all the output files generated by GROMACS analysis tools into a single HDF file using the following command:

```
md_davis collect \
--backbone_rmsd rmsd.xvg --backbone_rg rg.xvg \
--trajectory trajectory.trr --structure structure.gro
--rmsf rmsf.xvg 0 500 \
--ss dssp.dat \
--sasa resarea.xvg \
--info information.json \
output1.h5
```

If the `--trajectory` and `--structure` options are provided. MD DaVis will calculate the backbone dihedral angles for all frames and the circular standard deviation of each dihedral angle.

Note the numbers at the end of the `--rmsf` options are the start and end time for the RMSF calculation in nanosecond. These will be inserted as attributes in the HDF file and must be provided. In case, the RMSF for each chain was calculated separately, the files may be provided to `--rmsf` option in the correct order followed by the start and end times.

Additional details are available with `-h` option for each MD DaVis command, such as

```
md_davis collect -h
```

Step 3: Plotting overlaid residue data **Step 3a:** Create a pickle file with the residue dataframe using:

```
md_davis residue dataframe --prefix name1 output1.h5 data1.p
```

The optional argument `-a annotations.json` can be provided to place a mark at certain residue locations. The contents of `annotations.json` should be of the following form:

```
{
    "chain 0": {"Active Site": [23, 41], "Substrate Binding Site": [56]},
    "chain 1": {"Nucleotide Binding Regions": [15, 18]}
}
```

Each type of annotation is rendered with a different mark. Following annotations are available at present: * Active Site * Nucleotide Binding Regions * NADP Binding Site * Substrate Binding Site * Metal Binding Site * Cofactor Binding Site * Mutation

Step 3b: If your proteins are of different lengths and you need the peaks to be aligned, create a JSON file as shown below.

```
{
    "alignment": "path/to/alignment_file.clustal_num",
    "locations": {
        "name1": "name1_residue_wise_data.p",
        "name2": "name2_residue_wise_data.p",
        "name3": "name3_residue_wise_data.p"
    },
    "output": "acylphosphatase_residue_wise_data_aligned.p"
}
```

The contents of the alignment file, `alignment_file.clustal_num` must be in CLUSTAL format; for example:

```
CLUSTAL O(1.2.4) multiple sequence alignment

name1      --STARPLKSVDYEVFGRVQGVCFRMYAEDEARKIGVVGWVKNTSKGTVTGQVQGPEEKV     58
name2      --------PRLVALVKGRVQGVGYRAFAQKKALELGLSGYAENLPDGRVEVVAEGPKEAL     52
name3      ---VAKQIFALDFEIFGRVQGVFFRKHTSHEAKRLGVRGWCMNTRDGTVKGQLEAPMMNL     57
                     : *:**** :*  .  :. .  : *:  *   * *     .    :

name1      NSMKSWLSKVGSPSSRIDRTNFSNEKTISKLEYSNFSVRY 98
name2      ELFLHHLKQ--GPRLARVEAVEVQWGEE--AGLKGFHVY- 87
name3      MEMKHWLENNRIPNAKVSKAEFSQIQEIEDYTFTSFDIKH 97
            :   :     *          :           * :
```

Step 3b: Plot the residue data pickle file from the previous command using:

```
md_davis plot residue data1.p data2.p
```

Step 4: Free energy Landscapes

Create and plot free energy landscapes using common bins and ranges

```
md_davis landscape rmsd_rg -T 300 --common --select backbone output1.h5 output2.h5 -s␣
↪landscapes.h5
```

This command will create an html file with the interactive landscapes. It will not open the file like other plotting commands, so check the working directory for the output html file. ### Plot free energy landscape overlaid with trajectory points One must save the landscape created by the previous command with `-s` before this one can be used. Since the output generated for single landscape is big, visualization of multiple landscapes becomes impractical. So, it only plots one landscape at a time. Select the desired landscape in `landscapes.h5` by providing its index with `-i`. By default only the first landscape is plotted

```
md_davis landscape animation landscapes.h5 -i 0 --static -o trajectory.html
```

1.2.2 Install

System Requirements

- A 64-bit operating system
- A Python 3 installation with version 3.7

Conda Installation

The easiest method to install is with Anaconda or Miniconda, which works on all operating systems. It is highly recommended to install MD DaVis in a virtual environment. The environment.yml is provided to ease the process.

```
conda env create -f environment.yml -n md_davis_env
```

This automatically creates a conda environment called `md_davis_env` with all required dependencies. Activate the environment and install MD DaVis in it using:

```
conda activate md_davis_env
pip install md-davis
```

If you don't have pip installed, this Python installation guide can guide you through the process.

Finally, obtain the Delphi and MSMS if you intend to perform electrostatics analysis.

Installation in the base conda environment

If you do not want to use the environment.yml file or if you want to install it in the base environment. Install mdtraj and pymol before installing md-davis.

```
conda install -c conda-forge mdtraj pymol-open-source
pip install md-davis
```

Linux Installation

MD DaVis can be installed on Linux with:

```
pip install md-davis
```

Note: PyMOL is required to run MD DaVis, which may be challenging to install in a virtual environment without `conda`. (see *External Dependencies*)

Windows Installation

On Windows, `pip` may fail to install MD DaVis due to errors with compiling dependencies. Please follow the instructions under *Conda Installation*.

Development Version

You can obtain the source code by cloning the public repository:

```
git clone git@github.com:djmaity/md-davis.git
```

or downloading the tarball:

```
curl -OL https://github.com/djmaity/md-davis/tarball/master
```

Once you have a copy of the source, it can be installed with:

```
python setup.py install
```

OR

```
pip install path/to/extracted/source/code
```

You can also install the development version directly with:

```
pip install https://github.com/djmaity/md-davis/archive/master.zip
```

External Dependencies

GROMACS

Currently, most analyses have to be performed with GROMACS, and the output is provided to MD DaVis for visualization. Other analysis tools may be used as long as the input to MD DaVis can be appropriately formatted.

PyMOL

PyMOL is not available in the python package index. Therefore, it cannot be automatically installed with `pip`.

Open-Source PyMOL

Open-Source PyMOL available from conda-forge can be installed with:

```
conda install -c conda-forge pymol-open-source
```

Alternatively, on Linux, PyMOL can be installed with the system package manager, e.g., `apt` in Ubuntu or `dnf` in Fedora. However, it is not possible to install PyMOL into a virtual environment using this method. Therefore, MD DaVis must be installed in the system python as well.

On Windows, if you are not using `conda`, then pre-built Open-Source PyMOL can be downloaded from Christoph Gohlke's page distributing unofficial windows binaries for python extension packages.

Commercial/Educational PyMOL

Commercial/Educational PyMOL from Schrödinger can be installed with:

```
conda install -c schrodinger pymol-bundle
```

DelPhi and MSMS

Python dependencies are automatically installed. However, the electrostatics calculation requires the following two programs, which must be obtained separately.

- Delphi
- MSMS

1.2.3 Uninstall

MD DaVis can be easily uninstalled like any other python package, with:

```
pip uninstall md-davis
```

As with any python package, this does not remove the dependencies installed by MD DaVis. That is why it is recommended to install MD DaVis in a virtual environment. Then, the virtual environment may be entirely removed without affecting other python packages on the system.

Note: On Linux, if MD DaVis was installed as root or with `sudo`, the uninstall command should be run with `sudo`.

1.2.4 Free Energy Landscapes

MD DaVis can create alchemical free energy landscapes using two variables and multiple such landscapes can be plotted together. The landscapes can be plotted using common ranges and same number of bins. The wonderful feature of the plot created using MD DaVis is that it is an interactive html, and rotating one subplot rotates all others to the same view. These features facilitate the quick and efficient comparison of the free energy landscapes.

The free energy landscape of a protein is a continuous function between the protein conformations and the free energy in which a limited number of collective variables represents each conformation. MD DaVis creates free energy landscapes using the method described in Tavernelli et al., 2003. First, the user must provide two quantities of interest to classify the protein conformations, such as RMSD and RG, or the projection on the first two eigenvectors from principal component analysis. Then, the two variables are binned to make a 2D histogram, and the free energy landscape is obtained by Boltzmann inversion of the histogram using:

$$\Delta\varepsilon_i = -k_B T \ln\left(\frac{n_i}{n_{max}}\right)$$

where, $\Delta\varepsilon_i$ is the change in free energy between the conformations in the ith bin and the maximally occupied bin, n_i and n_{max} are the number of conformations in these, respectively. This formula results in 0 energy for the most probable state and all other states having positive energy. Since the absolute free energy is impossible to calculate and only the change in free energy is meaningful, the maximum value in the free energy landscape is subtracted from all the bins so that the most probable state has the most negative free energy. An excellent feature of MD DaVis is that while creating the 2D histogram, it creates a data structure in which the index of each frame in a particular bin is stored. This feature allows one to find the frames in a particular bin of the landscape, and the frames in the minima are generally of

particular interest. Multiple free energy landscapes can be calculated and plotted, ensuring that the range and binning of each axis are shared across all the landscapes evaluated together, enabling their direct. The interactive 3D plot of the free energy landscape can be rotated and viewed from various directions, zoomed in and out, and labels on the surface inspected to determine the valleys and wells, which is not possible in static 3D plots or contour plots. A unique feature of MD DaVis is that rotating one landscape in a plot with multiple landscapes rotates all to the same view. This feature of synchronizing the view of 3D plots is not available in any other plotting library. The trajectory points can also be animated or plotted on top of the free energy landscape; this allows inspecting the traversal of well in the free energy landscapes during the MD simulation.

Note: Plotting free energy landscapes requires a large number of frames, therefore, always save enough number of frames during the simulation. The number of frames required to obtain a reasonable representation of the free energy landscape depends on flexibility of the protein. If the protein accesses a large number of conformational states a greater number of frames would be required. As a ball park, at least 10000 frames are required.

Warning: These commands do not open the output html file in a web browser like other plotting commands, so please check the working directory for the output html file. If a file with the same name exists, it will be overwritten.

Free Energy Landscapes from xvg files

The quickest way to plot the free energy landscape is from the .xvg files output by GROMACS.

```
md_davis landscape_xvg -c -T 300 -x rmsd_file_1.xvg -y rg_file_1.xvg -n "name_1" -l
↪"Free Energy Landscape for 1"
```

Here, the RMSD and $R_G$precalculated using GROMACS is provided as the `-x` and `-y` input files. The .xvg files are space or tab delimited text files with time and data in the first and second columns, respectively. The output from any other analysis tool may be used as input once the file is appropriately formatted. Therefore, other properties may be plotted on the x and y axes, for example, the first and the second principal components from a principal component analysis.

If the output filename is not provided the default name is `landscape.html`.

`-c` specifies that the ranges of the all the landscapes must be the same

`-T 300` specifies to use 300 K as the temperature for the boltzmann inversion. If this option is not provided, only the 2D histrogram of the data is plotted.

To plot multiple free energy landscapes as subplots just provide the inputs one after the other. The `-x`, `-y`, `-n`, and `-l` for one trajectory must be provided together before the next set of inputs from the other trajectory.

```
md_davis landscape_xvg -c -T 300 -x rmsd_file_1.xvg -y rg_file_1.xvg -n "name_1" -l
↪"Free Energy Landscape for 1" -x rmsd_file_1.xvg -y rg_file_1.xvg -n "name_2" -l "Free␣
↪Energy Landscape for 2" -x rmsd_file_1.xvg -y rg_file_1.xvg -n "name_3" -l "Free␣
↪Energy Landscape for 3"
```

Plot Free Energy Landscapes from HDF Files

If the RMSD and $R_G$have been collated into HDF files for each trajectory. These may be plotted using:

```
md_davis landscape rmsd_rg -c -T 300 file1.h5 file2.h5 file3.h5
```

Plot Free Energy Landscapes Overlaid with Trajectory Points

One must save the landscape created by `landscape_xvg` or `landscape` with the `-s` option before this one can be used. Since the output generated for single landscape is big, visualization of multiple landscapes becomes impractical. So, it only plots one landscape at a time. Select the desired landscape in `landscapes.h5` by providing its index with `-i`. By default only the first landscape is plotted

```
md_davis landscape animation landscapes.h5 -i 0 --static -o trajectory.html
```

Step 4: Free energy Landscapes

Create and plot free energy landscapes using common bins and ranges

```
md_davis landscape rmsd_rg -T 300 --common --select backbone output1.h5 output2.h5 -s␣
↪landscapes.h5
```

This command will create an html file with the interactive landscapes. It will not open the file like other plotting commands, so check the working directory for the output html file. ### Plot free energy landscape overlaid with trajectory points One must save the landscape created by the previous command with `-s` before this one can be used. Since the output generated for single landscape is big, visualization of multiple landscapes becomes impractical. So, it only plots one landscape at a time. Select the desired landscape in `landscapes.h5` by providing its index with `-i`. By default only the first landscape is plotted

```
md_davis landscape animation landscapes.h5 -i 0 --static -o trajectory.html
```

1.2.5 Surface Electrostatic Potential Per Residue

Usually, the distribution of electrostatic potential around a protein mole-cule is visualized by solving the Poisson-Boltzmann equation and color-ing the molecular surface with obtained electrostatic potentials, which can be compared qualitatively. MD DaVis provides a wrapper to calcu-late the electrostatic potential on the vertices of a triangulated molecular surface and extract the total and mean surface electrostatic potential per residue, enabling a quantitative comparison. DelPhi versions later than 8.0 (Li et al., 2019) can calculate the electrostatic potential at user-specified points, which was leveraged to calculate the surface electrostat-ic potential at each residue. The steps involved are as follows: 1. Calculate the triangulated molecular surface using the MSMS program (Sanner et al., 1996). 2. The electrostatic potentials at these vertices of the triangulated surface are calculated using DelPhi. 3. The vertices corresponding to each residue are identified along with the electrostatic potentials at these points. 4. The total and mean of the electrostatic potentials on the vertices for each residue are calculated. The total gives information about the charge on the surface as well as the exposed surface area, while the mean nullifies the contribution from the exposed surface area. This calculation is repeated for every confor-mation in a sample from the trajectory with tens or hundreds of frames. The mean and the standard deviation of both total surface potential and the mean surface potential are included in the overlaid residue plot creat-ed by MD DaVis. Using a sample from the MD for the electrostatics calculations incorporates the dynamical information into the plot. There-fore, it has greater confidence compared to inferences based on a single structure or frame. It is advisable to align the frames and use the same grid size for each Delphi calculation. However, the variation in the mean and total surface electrostatic potentials due to the different orientations of the protein molecule in the grid is negligible as long as the triangulated molecular surface is high resolution. The default parameters are used for running Delphi with 2000 linear interations and a maximum change in potential of 10-10 kT/e. The salt concentration is set to 0.15 M, and a solvent dielectric value of 80 (die-lectric of water) was used. The atomic

charges and radii from the CHARMM force field are provided by default. The output potential map is saved in the cube file format used by Gaussian software; the other formats do not load in molecular visualization software. A discrepancy between the molecular surface calculated by MSMS and the molecular surface used by DelPhi to detect the protein's interior was observed for most buried residues in multimeric proteins. DelPhi uses a different die-lectric constant for the molecule's interior, so the potential on these resi-dues was unreasonably high. This caveat should be borne in mind while analyzing the results of these calculations.

Delphi and MSMS must be installed for this to work. You may download these from: [Delphi](http://compbio.clemson.edu/delphi) and [MSMS](http://mgl.scripps.edu/people/sanner/html/msms_home.html)

You will need the parameter files for delphi run, which can be downloaded from [paramter files](http://compbio.clemson.edu/downloadDir/delphi/parameters.tar.gz)

Step 1: Calculate the electrostatics using the following command

```
md_davis electrostatics --surface_potential \
    -m ~/.opt/msms/ \
    -d ~/.opt/delphi8.4.2_sequential
    -c path/to/charge/file.crg \
    -r path/to/radius/file.siz \
    PDB_FILE.pdb \
    path/to/electrostatics/output/directory
```

The following charge and radius files are provided with the source code for MD DaVis.

```
md_davis/md_davis/electrostatics/charmm.crg
md_davis/md_davis/electrostatics/charmm.siz
```

If you receive a warning during Delphi run regarding missing charge or radius. Then the missing properties must be added to these files or whichever files you provide to *md_davis electrostatics*.

Step 2: Add the surface electrostatics data into the residue dataframe by modifying the command for creating residue dataframe to include the option *-d* and specifying the path to the directory containing the surface potential files. This will search for all *.pot* files and include their mean and total in the residue dataframe.

```
md_davis residue dataframe \
        --potential  path/to/electrostatics/output/directory \
        --annotation  annotations.json \
        --prefix  prefix \
        md_davis_collect_output_data.h5  \
        output_residue_wise_data.p
```

Step 3: Plot the residue dataframe with the usual command:

```
md_davis plot residue output_residue_wise_data.p
```

The electrostatic potentials calculated for a sample of frames from the MD trajectories, as discussed above, can be visualized as a 3D animation of electric field lines (Figure 1C). This highlights the effect of dynamics of the electric field in the vicinity of the molecule. The 3D electric field dynamics is visualized as described below: 1. The coordinates of the reference structure are translated to place the center of mass of the molecule at the origin and rotated so that the first, second, and third principal axes are along the x, y, and z-axes, respectively. 2. The frames sampled from the trajectory are aligned to the refer-ence. 3. The electrostatic potentials are obtained for each sampled struc-ture using Delphi. The box for each calculation is centered at the origin, and the number of grid points is manually set to the same value for each structure to ensure the same box size during each calculation. 4. The surface electrostatic potentials calculated per residue or atom are written into the output PDB file's B-factor or occupancy col-umn. 5. The output PDB file and the corresponding electric field from the sample are visualized as frames in PyMOL (Schrödinger, LLC, 2015), which can animate the dynamics of the electric field lines.

1.2.6 Electric Field Dynamics

1.2.7 Residue Property Plot

Understanding the function–dynamics relationship of protein often re-quires comparing two or more properties, e.g., the effect of dihedral fluctuation on the RMSF of the protein. The two properties, dihedral fluctuation and RMSF, can be easily plotted with any plotting program. Later, one might be interested in knowing the trend between the RMSF and solvent exposure of the residues. Traditionally, each time new prop-erties are compared, new plots need to be prepared. This repetitive pro-cess is alleviated by making overlaid plots of all the residue properties at once (Figure 1A). MD DaVis can plot the following quantities: RMSF, torsional flexibility, secondary structure, solvent accessible surface area, and surface electrostatic potential on each residue. Showing all the prop-erties as an overlaid plot can be overwhelming and cluttering. However, the option to interactively turn on or off the visualization for each data series clears the clutter and highlights the similarities and differences. Moreover, the labels and annotations that appear on hovering the cursor over certain regions improve the interpretability of these plots, thereby granting substantial time savings. The data obtained from multiple trajectories can be overlaid in a single interactive plot for ease of comparison, which removes the need for making multiple plots and immediately brings out the distinguishing features between the trajectories. The novel feature of MD DaVis is that it can align the data for similar proteins by inserting required gaps along the x-axis using an alignment file. Aligning the residues on the x-axis aligns the peaks, highlighting the similarities between the datasets, which is incredibly powerful when comparing the dynamical information from similar proteins.

4. **Create interactive plot of containing the following residue level properties:**
    - root mean squared fluctuation (RMSF)
    - torsional flexibility (circular standard deviation of backbone dihedral angles)
    - Secondary structure
    - Solvent accessible surface area
    - Mean and standard deviation for the total surface electrostatic potential per residue
    - Mean and standard deviation for the mean surface electrostatic potential per residue

It can also use an alignment to align the residue level data from different proteins along the x-axis. This ensures that the peaks line up properly for better interpretation.

Traditionally, each time the analysis of MD trajectories are compared new statis plots have to be created.

Conventional analysis requires plotting the data each time new properties are compared. This repetitive process is alleviated by making overlaid plots of all the informative residue propertie

---

Note: the paths in the input toml file is relative to the location where the md_davis command will be called from. To avoid any confusion try using absolute paths.

How to interact with the plot

Step 3: Plotting overlaid residue data **Step 3a:** Create a pickle file with the residue dataframe using:

```
md_davis residue dataframe --prefix name1 output1.h5 data1.p
```

The optional argument `-a annotations.json` can be provided to place a mark at certain residue locations. The contents of `annotations.json` should be of the following form:

```
{
    "chain 0": {"Active Site": [23, 41], "Substrate Binding Site": [56]},
    "chain 1": {"Nucleotide Binding Regions": [15, 18]}
}
```

Each type of annotation is rendered with a different mark. Following annotations are available at present: * Active Site * Nucleotide Binding Regions * NADP Binding Site * Substrate Binding Site * Metal Binding Site * Cofactor Binding Site * Mutation

Step 3b: If your proteins are of different lengths and you need the peaks to be aligned, create a JSON file as shown below.

```
{
    "alignment": "path/to/alignment_file.clustal_num",
    "locations": {
        "name1": "name1_residue_wise_data.p",
        "name2": "name2_residue_wise_data.p",
        "name3": "name3_residue_wise_data.p"
    },
    "output": "acylphosphatase_residue_wise_data_aligned.p"
}
```

The contents of the alignment file, `alignment_file.clustal_num` must be in CLUSTAL format; for example:

```
CLUSTAL O(1.2.4) multiple sequence alignment

name1      --STARPLKSVDYEVFGRVQGVCFRMYAEDEARKIGVVGWVKNTSKGTVTGQVQGPEEKV     58
name2      --------PRLVALVKGRVQGVGYRAFAQKKALELGLSGYAENLPDGRVEVVAEGPKEAL     52
name3      ---VAKQIFALDFEIFGRVQGVFFRKHTSHEAKRLGVRGWCMNTRDGTVKGQLEAPMMNL     57
                     : *:**** :*  .  :. .  : *:  *   * *     .    :

name1      NSMKSWLSKVGSPSSRIDRTNFSNEKTISKLEYSNFSVRY 98
name2      ELFLHHLKQ--GPRLARVEAVEVQWGEE--AGLKGFHVY- 87
name3      MEMKHWLENNRIPNAKVSKAEFSQIQEIEDYTFTSFDIKH 97
            :   :     *          :           * :
```

Step 3b: Plot the residue data pickle file from the previous command using:

```
md_davis plot residue data1.p data2.p
```

Annotations

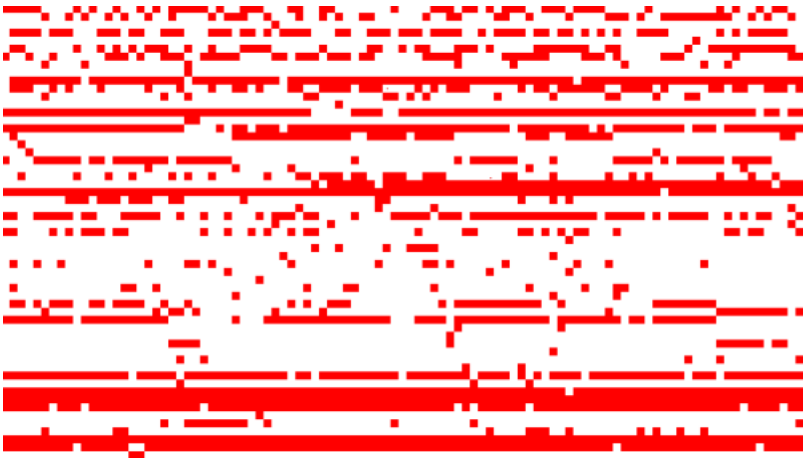
```
{
    "chain 0": {"Active Site": [23, 41], "Substrate Binding Site": [56]},
    "chain 1": {"Nucleotide Binding Regions": [15, 18]}
}
```

Each type of annotation is rendered with a different mark. Following annotations are available at present: * Active Site * Nucleotide Binding Regions * NADP Binding Site * Substrate Binding Site * Metal Binding Site * Cofactor Binding Site * Mutation

1.2.8 Contact/H-bond matrix

Motivation

GROMACS has an excellent tool to calculate hydrogen bonds (H-bonds) or contacts from trajectories called gmx hbond. It is written in C++, and the loops are parallelized. Therefore, its performance cannot be matched by any Python code. The default output from gmx hbond is a `.xvg` file containing the total number of H-bonds/contacts between the selected groups over all frames, which is not very informative. The option `-hbm` outputs the existence matrix for all H-bonds/contacts over all frames in an X PixMap (`.xpm`) image:



Only a few image viewers support `.xpm` file format, and the image cannot be easily interpreted because it does not contain any labels for the participating atoms and residues in the H-bonds/contacts. To overcome these limitations, use MD DaVis for analyzing the output files created by `gmx hbond`.

Steps

1. Calculate the H-bonds/contacts using `gmx hbond` with the options `-hbm` and `-hbn`:

```
gmx hbond -f trajectory.xtc -s structure.tpr -num hbnum.xvg -hbm hb_matrix -hbn hb_index
```

To calculate the contacts provide `--contact` option. `gmx hbond` requires the `.tpr` file to be supplied to the `-s` option, and other structure formats like `.pdb` or `.gro` are not accepted. The order of rows in the hb_matrix.xpm image is the same as the order of atoms in the hb_index.ndx file. MD DaVis utilizes this to determine the atoms in each H-bond/contact (rows in the `.xpm` file).

2. Create and save the H-bonds/contacts data:

```
md_davis hbond -f hb_matrix.xpm --index hb_index.ndx -s structure.tpr --pickle hb_data.p␣
↪-g hbonds_Protein
```

3. Plot the H-bonds/contacts matrix:

```
md_davis plot_hbond --percent --total_frames 101 --cutoff 33 -o 2VH7_hbond_matrix.html␣
↪2VH7_hbonds.p
```

The above command plots the percentage of the H-bonds, which is calculated for each H-bond as follows:

number of frames the H-bond is observed / total number of frames * 100

The cutoff of 33 % is set to plot only those H-bonds whose occurrence is greater than 33 %.

Bonus Tip

Scroll down the hb_index.ndx file to find the name of the group containing the atoms participating in H-bonds/contacts. It contains three or two columns of data for H-bonds or contacts, respectively.

```
 1235  1243  1244  1248  1249  1260  1261  1263  1264  1280  1285  1286  1301  1302  1320
 1321  1339  1340  1356  1361  1362  1380  1381  1389  1390  1392  1393  1410  1413  1414
 1421  1424  1425  1433  1434  1436  1437  1456  1457  1468  1469  1473  1474  1492  1493
 1508  1509  1525  1530  1531
[ hbonds_Protein ]
      9     10     35
     36     37    773
     55     56   1285
     62     63    725
     62     63    726
```

Hydrogen bonds (H-bonds) and contacts in a protein or between protein and ligand are calculated to understand the interactions essential for its function. GROMACS has the hbond utility to calculate the H-bonds and contacts purely from geometric considerations. By default, the command only returns the number of H-bonds/contacts detected in each frame, which is not informative enough. However, by providing suitable options to the hbond utility, an index file with the list of the atoms involved in each H-bond/contact detected and a matrix with the H-bonds/contacts can be obtained. The matrix has H-bonds/contacts listed in the index file along one dimension and the trajectory frames along the other dimension. The elements of the matrix denote the presence of a hydrogen bond at a particular frame. Unfortunately, the matrix does not have any labels, and its size becomes huge for long simulations making it uninterpretable. MD DaVis processes these files to calculate the number of frames with the H-bond/contact, which can be converted to a percentage. This can then be represented as a matrix where the participating atoms are along the two dimensions of the matrix (Figure 1D).

1.2.9 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

Types of Contributions

Report Bugs

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

MD DaVis could always use more documentation, whether as part of the official MD DaVis docs, in docstrings, or even on the web in blog posts, articles, and such.

Feature Requests and Feedback

The best way to send feedback is to file an issue.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Development

Create the development environment and install the dependencies using the dev_environment.yml file. This installs packages for linting, packaging and building documentation in addition to the core dependencies.

```
conda env create -f dev_environment.yml -n md_davis_dev
conda activate md_davis_dev
```

Install md-davis in editable mode:

```
pip install -e md-davis
```

1.2.10 History

0.1.0 (2019-09-06)

- First release on PyPI.

0.2.0 (2020-04-10)

- Second alpha release on GitHub.

1.3 Tutorials

1.3.1 Acylphosphatase Homologs

In this tutorial, we shall analyze the molecular dynamics (MD) trajectories of four acylphosphatase (AcP) homologs:

PDB ID	Organism
2VH7	Human
2GV1	E. coli
2ACY	Bovine
1URR	Fruit Fly

If you are new to running MD simulations using GROMACS, please refer to the legendary GROMACS tutorial Lysozyme in Water by Dr. Justin A. Lemkul at http://www.mdtutorials.com/.

RMSD and R_G

Often the first step after a successful MD simulation is to calculate the root-mean-square deviation (RMSD) and radius of gyration (R_G). In GROMACS, these can be calculated using gmx rms and gmx gyrate, respectively.

```
gmx rms -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_rmsd
gmx gyrate -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_rg
```

2VH7_rmsd.xvg and 2VH7_rg.xvg are text files containing the RMSD and R_G, respectively. Repeat the process for all the other trajectories.

Note: If you have installed MD DaVis in a virtual or conda environment as suggested in the installation instructions, make sure to activate it before running the `md_davis` commands.

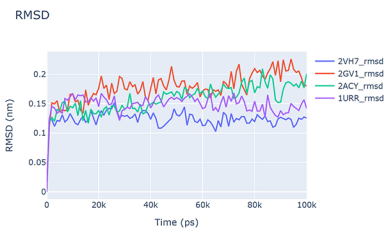
Plot 2VH7_rmsd.xvg using:

```
md_davis xvg 2VH7/2VH7_rmsd.xvg -o 2VH7/2VH7_rmsd.html
```

You should obtain a plot like this:

To plot the RMSD from the four trajectories together, use:

```
md_davis xvg 2VH7/2VH7_rmsd.xvg 2GV1/2GV1_rmsd.xvg 2ACY/2ACY_rmsd.xvg 1URR/1URR_rmsd.xvg␣
↪-o AcP_rmsd.html
```



Similarly, the R_G can also be plotted using the `md_davis xvg` command.

Create Free Energy Landscapes

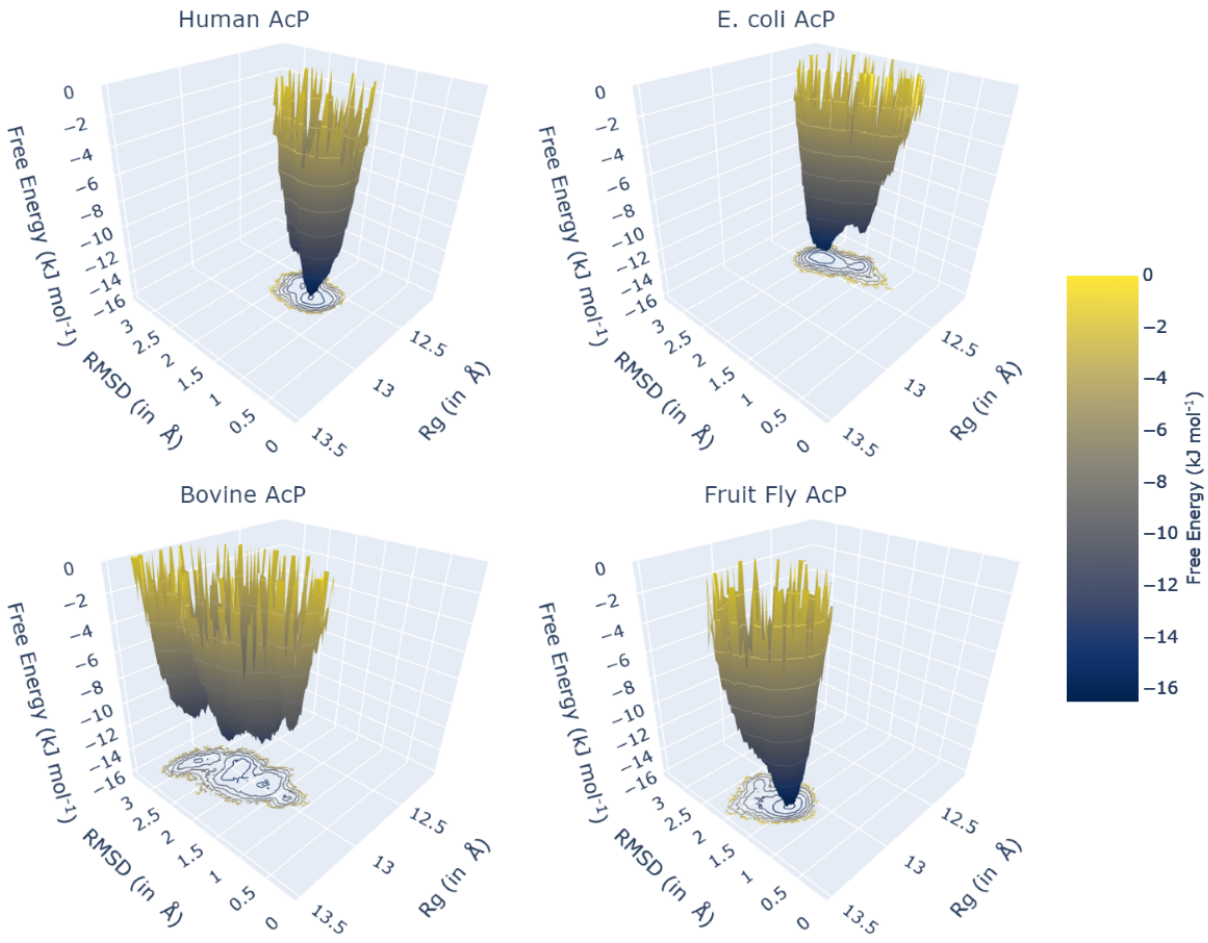
Next, we will create the free energy landscape using RMSD and R_G as the *x* and *y* variables. The sample trajectories provided with this tutorial only contain 100 frames to keep their sizes small. Thus, the RMSD and R_G files created at the beginning of this tutorial only contain 100 timesteps each. Using 100 points to create a free energy landscape would not be accurate. Therefore, the RMSD and R_G files calculated from the full 1 s trajectory containing 100,000 frames are provided with the tutorial. Use these files to create the free energy landscape. For human acylphosphatase, the free energy landscape can be created with:

```
md_davis landscape_xvg -T 300 -x 2VH7/2VH7_rmsd_full.xvg -y 2VH7/2VH7_rg_full.xvg -n
↪"2VH7" -l "Human AcP" -o 2VH7_landscape.html
```

Here, `-T 300` specifies 300 K as the temperature of the system. Now to plot all four landscapes together:

```
md_davis landscape_xvg -T 300 --common -x 2VH7/2VH7_rmsd_full.xvg -y 2VH7/2VH7_rg_full.
↪xvg --name "2VH7" --label "Human AcP" -x 2GV1/2GV1_rmsd_full.xvg -y 2GV1/2GV1_rg_full.
↪xvg --name "2GV1" --label "E. coli AcP" -x 2ACY/2ACY_rmsd_full.xvg -y 2ACY/2ACY_rg_
↪full.xvg --name "2ACY" --label "Bovine AcP" -x 1URR/1URR_rmsd_full.xvg -y 1URR/1URR_rg_
↪full.xvg --name "1URR" --label "Fruit Fly AcP" -o AcP_FEL.html
```

In the command above, remember to provide `-x`, `-y`, `--name`, and `--label` together before those for the subsequent trajectory. The option `--common` instructs MD DaVis to create the four landscapes using identical ranges and binning, which allows us to compare the landscapes reliably. The output from the above command is shown below; click the image to view the interactive HTML file.



Electrostatic Potential and Electric Field Dynamics

1. Create a sample of frames for calculating the electrostatic potential with DelPhi

```
mkdir 2VH7/2VH7_electrostatics/
gmx trjconv -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_
↪electrostatics/2VH7_frame.pdb -dt 10000 -sep
```

2. MD DaVis has the `electrostatics` command, which is a wrapper for running DelPhi and reporting the electrostatic potential at the vertices of a triangulated surface obtained using MSMS

```
md_davis electrostatics --surface -m ~/msms_i86_64Linux2_2.6.1/msms.x86_64Linux2.2.6.1 -
↪d ~/delphicpp_v8.4.5_serial -o 2VH7/2VH7_electrostatics/ 2VH7/2VH7_electrostatics/2VH7_
↪frame*.pdb
```

In the command above, the MSMS directory and the DelPhi executable are placed in the home folder. Adjust the path according to your system.

3. The electrostatic potential on the surface and the dynamics of the electric field around the molecule can be visualized with the following command:

```
md_davis electrodynamics --ss_color --surface --name Human_AcP 2VH7/2VH7_electrostatics
```

Residue Properties Plot

1. Calculate the root-mean-square fluctuation, solvent accessible surface area, and secondary structure using GROMACS:

```
gmx rmsf -res -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_rmsf
gmx sasa -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_sasa.xvg -
↪or 2VH7/2VH7_resarea.xvg
gmx do_dssp -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_structure.pdb -o 2VH7/2VH7_dssp -
↪ssdump 2VH7/2VH7_dssp -sc 2VH7/2VH7_dssp_count
```

Repeat for the remaining trajectories. We will also plot the torsional flexibility, but that will be calculated by MD DaVis later.

Note: For the `gmx do_dssp` command to work, the `dssp` or `mkdssp` binary must be available on your system. Download it from ftp://ftp.cmbi.ru.nl/pub/software/dssp/ and ensure GROMACS can find it by setting the `DSSP` environment variable to point to its location on your system.

---

2. Collect and store all the calculated properties into an HDF file. To do that, first, create a TOML file as shown below, telling MD DaVis the location of each file.

```
name = '2VH7'
output = '2VH7_data.h5'
label = 'Human AcP'
text_label = 'Human AcP'

trajectory = '2VH7_trajectory.xtc'
structure = '2VH7_structure.pdb'

[timeseries]
    rmsd = '2VH7_rmsd_full.xvg'
    rg = '2VH7_rg_full.xvg'

[dihedral]
    chunk = 101

[residue_property]
    secondary_structure = '2VH7_dssp.dat'
    sasa = '2VH7_resarea.xvg'
    surface_potential = '2VH7_electrostatics'   # directory containing electrostatic␣
↪calculations

    [residue_property.rmsf]
        rmsf_files = '2VH7_rmsf.xvg'
        start = 0
        end = 100
```

Input TOML file for each trajectory is provided with the tutorial files. Next, collate all the data using MD DaVis, which can process multiple TOML files and create the respective HDF file.

```
md_davis collate 2VH7/2VH7_input.toml 2GV1/2GV1_input.toml 2ACY/2ACY_input.toml 1URR/
↪1URR_input.toml
```

3. Combine the data from the HDF file into a pandas dataframe with:

```
md_davis residue 2VH7_data.h5 2GV1_data.h5 2ACY_data.h5 1URR_data.h5 -o AcP_residue_data.
↪p
```

4. Plot the residue properties:

```
md_davis plot_residue AcP_residue_data.p -o AcP_residue_data.html
```

Now, we can also align the residues of the different trajectories to align the peaks in the data.

1. obtain the sequence of residues in FASTA format from each PDB file using the `sequence` command in MD DaVis:

```
md_davis sequence 2VH7/2VH7_structure.pdb -r fasta
```

2. Use a sequence alignment program or webservers like Clustal Omega or T-coffee to obtain the alignment of these sequences in ClustalW format.

```
CLUSTAL O(1.2.4) multiple sequence alignment


2GV1_structure      ---MSKVCIIAWVYGRVQGVGFRYTTQYEAKRLGLTGYAKNLDDGSVEVVACGEEGQVEK    57
1URR_structure      -VAKQIFALDFEIFGRVQGVFFRKHTSHEAKRLGVRGWCMNTRDGTVKGQLEAPMMNLME    59
2VH7_structure      ----TLISVDYEIFGKVQGVFFRKHTQAEGKKLGLVGWVQNTDRGTVQGQLQGPISKVRH    56
2ACY_structure      AEGDTLISVDYEIFGKVQGVFFRKYTQAEGKKLGLVGWVQNTDQGTVQGQLQGPASKVRH    60
                          ..:   ::*:**** **  *. *.*:**: *:  *   *:*:    .   :: .

2GV1_structure      LMQWLKSGGPRSARVERVLSEPH--HPSGELTDFRIR-  92
1URR_structure      MKHWLENNRIPNAKVSKAEFSQIQEIEDYTFTSFDIKH  97
2VH7_structure      MQEWLETRGSPKSHIDKANFNNEKVILKLDYSDFQIVK  94
2ACY_structure      MQEWLETKGSPKSHIDRASFHNEKVIVKLDYTDFQIVK  98
                    : .**:.    .::::.:.         .   :.* *
```

3. Create a TOML file to specify which alignment file corresponds to which chain and which sequence label corresponds to which data, as shown below:

```
[names]
2GV1 = '2GV1_structure'
1URR = '1URR_structure'
2VH7 = '2VH7_structure'
2ACY = '2ACY_structure'

[alignment]
'chain 0' = 'AcP_alingment.clustal_num'
```

4. Run the `md_davis residue` command passing the TOML file with the `--alignment` option to generate the pandas dataframes.

```
md_davis residue 2VH7_data.h5 2GV1_data.h5 2ACY_data.h5 1URR_data.h5 --alignment Acp_
↪alignment_input.toml -o AcP_residue_data_aligned.p
```

5. Plot the aligned data frames.

```
md_davis plot_residue AcP_residue_data_aligned.p -o AcP_residue_data_aligned.html
```

Hydrogen Bond Matrix

1. Calculate the hydrogen bonds using the hbond utility in GROMACS.

```
gmx hbond -f 2VH7/2VH7_trajectory.xtc -s 2VH7/2VH7_md.tpr -num 2VH7/2VH7_hbnum.xvg -hbm␣
↪2VH7/2VH7_hb_matrix -hbn 2VH7/2VH7_hb_index
```

2. Open the output index file 2VH7_hb_index.ndx and scroll down to find the title of the last section containing the list of hydrogen bonds, which is hbonds_Protein in this case, as shown below:

```
 1235  1243  1244  1248  1249  1260  1261  1263  1264  1280  1285  1286  1301  1302  1320
 1321  1339  1340  1356  1361  1362  1380  1381  1389  1390  1392  1393  1410  1413  1414
 1421  1424  1425  1433  1434  1436  1437  1456  1457  1468  1469  1473  1474  1492  1493
 1508  1509  1525  1530  1531
[ hbonds_Protein ]
      9     10     35
     36     37    773
     55     56   1285
     62     63    725
```

3. Calculate the occurrence of each hydrogen bond:

```
md_davis hbond -x 2VH7/2VH7_hb_matrix.xpm -i 2VH7/2VH7_hb_index.ndx -s 2VH7/2VH7_
↪structure.pdb -g hbonds_Protein --save_pickle 2VH7/2VH7_hbonds.p
```

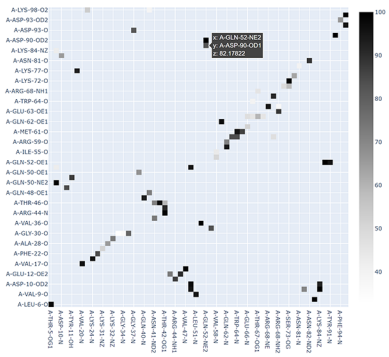
4. Plot the hydrogen bonds matrix

```
md_davis plot_hbond --percent --total_frames 101 --cutoff 33 -o 2VH7_hbond_matrix.html␣
↪2VH7/2VH7_hbonds.p
```

The above command plots the percentage of the H-bonds, which is calculated for each H-bond as follows:

number of frames the H-bond is observed / total number of frames * 100

The cutoff of 33 % is set to plot only those H-bonds whose occurrence is greater than 33 %.



1.3.2 Sickle and Normal Oxyhemoglobin

In this tutorial we shall compare the MD of sickle and normal oxyhemoglobin.

Creating the residue property plot is a bit more challenging because of the four chain in hemoglobin and the heme in each chain.

Defining the parameters in the input file to `md_davis collate` is critical in this case.

1.4 Commands

Following are the list of commands available in the md_davis commandline tool.

1.4.1 md_davis collate

All the data from a simulation are collected into a single HDF5 file for easy and organized storage. The benefit of using a common binary format like HDF5 is that data access is much faster than storing in text files and later many functions can be written in C, C++ or other languages

in dihedral in toml add chain_lengths to correct for discrepancy between HEO

It uses HDF5 file format to store the heterogeneous data obtained from consolidating and organizing the data from multiple calculations using the h5py Python module. HDF5 being an open binary format, allows users to open these files directly in C, C++, or FORTRAN pro-grams for added performance. Moreover, the data can be inspected with the HDF View GUI.

If file exists delete it

Step 2b: Provide this sequence in JSON file below, along with a few other properties. Note that for multi-chain proteins the sequence for each chain would be separated by a '/'.

```
{
    "label": "MD Simulation",
    "short_label": "MD",
    "html": "<i>MD Simulation</i>",
    "short_html": "<i>MD Simulation</i>",
    "protein": "protein name",
    "scientific_name": "some organism",
    "common_name": "common name",
    "sequence": "PUT/YOUR/SEQUENCE/HERE"
}
```

The most important property here is the **sequence**, which tells `md_davis collect` of the number of chains in the molecule and the number of residues in each chain. The **short_html** will determine the labels for the data in the final plots. This file is named `information.json` in the next command.

Step 2c: Collect all the output files generated by GROMACS analysis tools into a single HDF file using the following command:

```
md_davis collect \
--backbone_rmsd rmsd.xvg --backbone_rg rg.xvg \
--trajectory trajectory.trr --structure structure.gro
--rmsf rmsf.xvg 0 500 \
--ss dssp.dat \
```

(continues on next page)

(continued from previous page)

```
--sasa resarea.xvg \
--info information.json \
output1.h5
```

If the `--trajectory` and `--structure` options are provided. MD DaVis will calculate the backbone dihedral angles for all frames and the circular standard deviation of each dihedral angle.

Note the numbers at the end of the `--rmsf` options are the start and end time for the RMSF calculation in nanosecond. These will be inserted as attributes in the HDF file and must be provided. In case, the RMSF for each chain was calculated separately, the files may be provided to `--rmsf` option in the correct order followed by the start and end times.

Additional details are available with `-h` option for each MD DaVis command, such as

```
md_davis collect -h
```

1.4.2 md_davis electrodynamics

1.4.3 md_davis electrostatics

1.4.4 md_davis hbond

1.4.5 md_davis landscape_animate

1.4.6 md_davis landscape_xvg

1.4.7 md_davis plot_hbond

1.4.8 md_davis plot_residue

1.4.9 md_davis residue

1.4.10 md_davis sequence

Step 2a: Obtain the sequence of the protein from the PDB file used to start the simulation.

```
md_davis sequence structure_used_for_simulation.pdb
```

1.4.11 md_davis xvg

The GROMACS molecular dynamics software bundles numerous analysis tools. The output from these are in (.xvg) format to be viewed with xmgrace. Although xmgrace has some powerful tools for quick calculation of mean standard deviation, etc., the plots are cumbersome to deal with and difficult to compare multiple files.

MD DaVis provides the command `xvg` to plot such files using plotly or matplotlib.

It is generally intended that the when plotting multiple xvg files, they contain the same kind of data. Therefore, the titles and axes labels in the last supplied file are used.

MD DaVis can also plot multiple Grace (.xvg) files, which is the format for the output files from many GROMACS analysis tools. For example, an interactive plot with RMSD and RG from multiple trajectories can be created for quick comparison.

```
md_davis xvg <path/to/file.xvg>
```

Replace <*path/to/file.xvg*> with the location of your `.xvg` file.

Create interactive plot for time series data: root mean squared deviation (RMSD) and radius of gyration (R:subscript:G)

<iframe src="/_static/acylphosphatase_rmsg_rg.html" frameborder="0" width="100%" height="500px"></iframe>

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search