

Markov-state modeling of biomolecular systems pt. 2



Toni Giorgino

National Research Council of Italy

toni.giorgino@cnr.it



Master in Bioinformatics for Health Sciences
Barcelona, 26 Apr 2019

Instructions

- Open github.com
- GIT clone (or unzip)
giorginolab/Markov-Tutorial-BCN-2019
- This will copy the exercise files in your machine

Plan of the practice

1. We compute a MSM of a “toy model”:
trajectory in a 1-D potential (R language).
2. (Advanced) Move to a realistic trajectory set:
prothrombin:inhibitor binding
 - a. PLUMED projection
 - b. build a simple MSM in R like before
3. (Self-study) Repeat the analysis with a ready-made library, HTMD

I. Toy model in 1D

- Start the R environment
- See the “[Markov From Scratch](#)” document
- Data file needed: [data1.csv.gz](#)
 - [data10.csv.gz](#) is also available for experimenting, see comments

```
$ zless data1.csv.gz
2.3000000000000000000000e+01
2.3000000000000000000000e+01
2.4000000000000000000000e+01
2.5000000000000000000000e+01
2.6000000000000000000000e+01
2.6000000000000000000000e+01
2.6000000000000000000000e+01
```

Let's now follow the R notebook

2. Prothrombin:ligand example

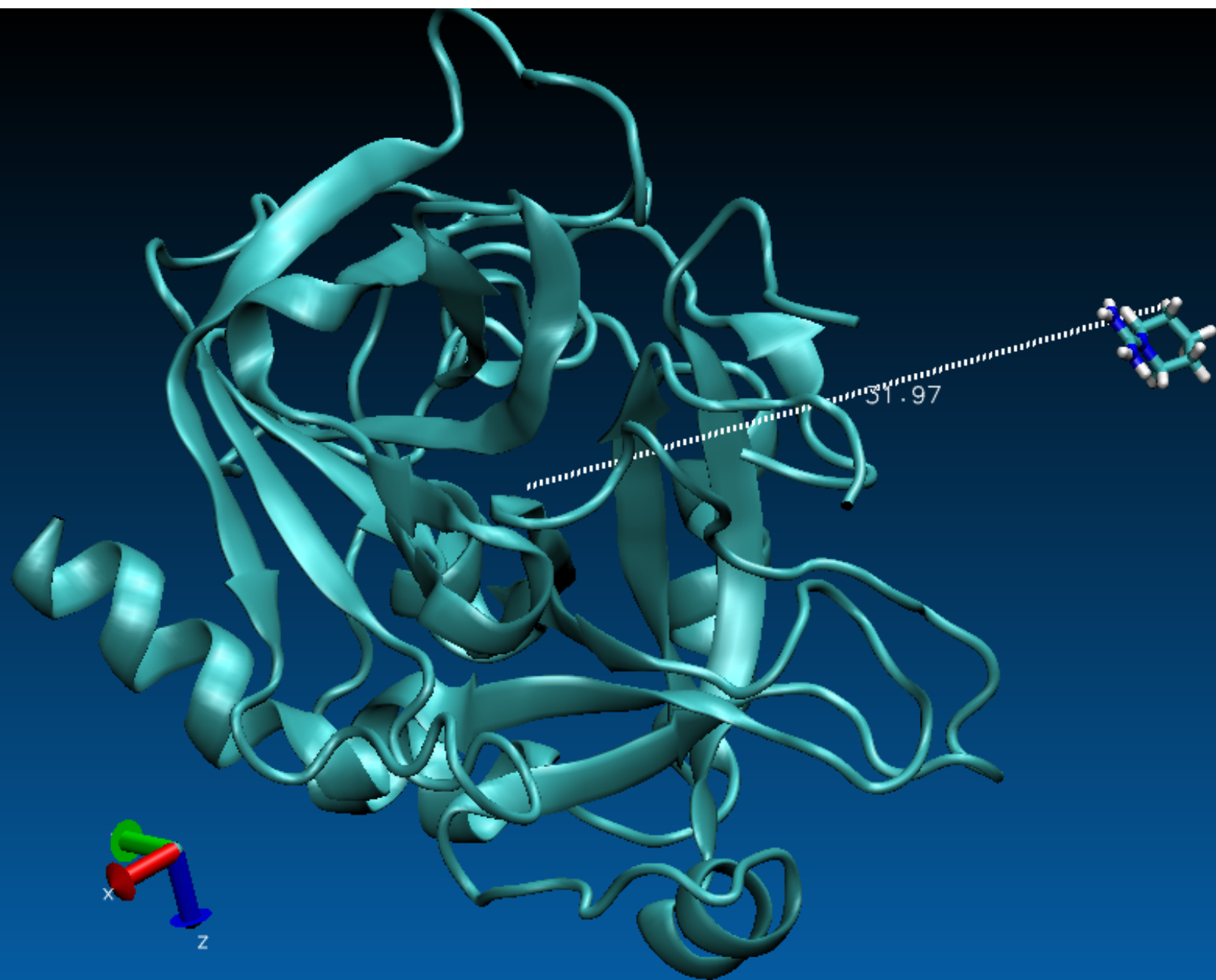
- This is a large, realistic simulation set
 - From htmd.org: [Ligand Binding Analysis](#)
 - 3 GB (don't download today)
 - 852 trajectories (in 3 groups)
 - $852 \times 20 \text{ ns} \approx 17 \text{ } \mu\text{s}$
- See the [*MarkovOnLargeDataset*](#) document

MarkovOnLargeDataset

- It is an R notebook similar to the I-D example. However...
- The original space is high-dimensional. So
 1. Project it with a PLUMED script:
Protein-Ligand vector after alignment
 2. Convert the 3D projection to discrete states
with a *k-means* clustering algorithm

Using PLUMED for projections

- We need to extract from *each frame of each trajectory* a limited number of values (projection, or metric)
- Must be orientation-independent (align)
- PLUMED may be used for the task
 - [protein-ligand-vector.plumed](#)
 - [project-with-plumed.sh](#)
- Results: [metric.dat](#)



protein-ligand-vector.plumed

```
UNITS LENGTH=A

FIT_TO_TEMPLATE STRIDE=1 REFERENCE=reference.pdb TYPE=OPTIMAL

# These are the serial numbers of protein CA's
prot: CENTER ...
ATOMS={ 5, 20, 30, 42, 52, 59, 78, 110, 116, 135, 155,
        170, 192, 214, 225, 244, 259, 271, 293, 307, 322, 346,
        . . .
        4229, 4253, 4272, 4294, 4316, 4340, 4359, 4376, 4398, 4414, 4433,
        4445, 4462
      }
...

# This is "resname MOL and noh" (inhibitors's heavy atoms)
ligand: CENTER ATOMS=4481,4484,4487,4490,4493,4496,4497,4498,4501

pl: DISTANCE ATOMS=prot,ligand COMPONENTS

PRINT ARG=pl.*
```

project-with-plumed.sh

Computes the metric over all the trajectories

(You don't actually have to run this. The results are already in **metrics.dat**)

```
#!/bin/bash

# This file computes the PLUMED metric indicated in $script
# on all of the trajectory files in $indir. The results
# are printed in stdout.

# You only need to run this file if you change the definition
# of the metric. The current results are in metric.dat

script=protein_ligand_vector.plumed
indir=/mnt/scratch/shared/markov/binding/1/filtered

# Loop over all files ending by .xtc
for tj in `find $indir -name \*.xtc -and -not -name .*`; do
    # Output file name
    outname=`basename $tj`
    echo Running plumed on $tj >&2
    plumed driver --plumed $script \
        --pdb $indir/filtered.pdb \
        --mf_xtc $tj | \
        egrep "^ " | sed "s+^+$outname+"
done
```

```
project-with-plumed.sh > newmetric.dat
```

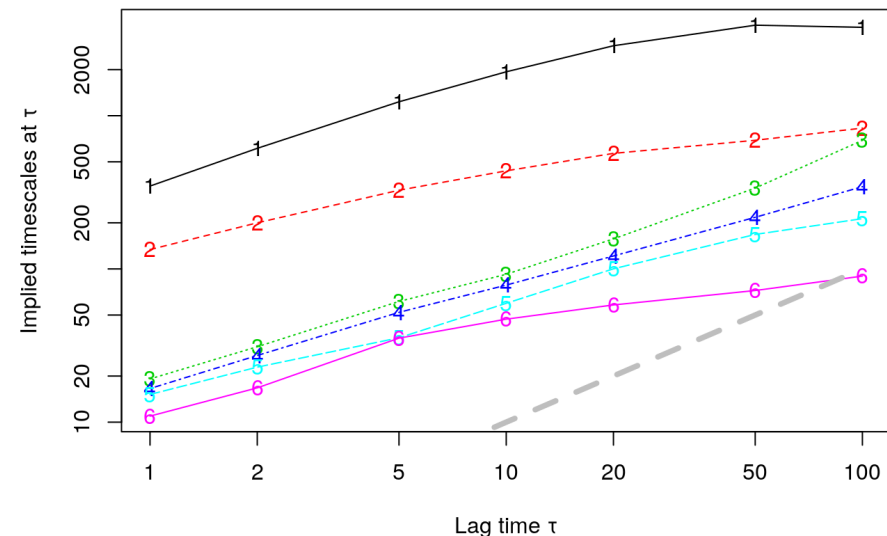
metric.dat

Results of the projection are in metric.dat.
This will be read in by R.

```
(htmd) tonigiorgino@monaco:~/practice$ head metrics.dat
e36s5_e27s3f37-....xtc 0.000000 -0.224091 -11.886612 1.973588
e36s5_e27s3f37-....xtc 1.000000 -0.413809 -12.038901 1.644167
e36s5_e27s3f37-....xtc 2.000000 -0.532973 -12.323975 2.144003
e36s5_e27s3f37-....xtc 3.000000 -0.300480 -11.363546 1.903810
e36s5_e27s3f37-....xtc 4.000000 -0.433555 -11.792027 2.057442
. . .
e24s3_e21s4f177-....xtc 196.000000 8.976495 -10.536748 2.060360
e24s3_e21s4f177-....xtc 197.000000 9.262883 -11.399111 2.169123
e24s3_e21s4f177-....xtc 198.000000 8.517335 -12.355884 2.464811
e24s3_e21s4f177-....xtc 199.000000 8.496182 -12.325769 1.489607
```

Where to go from here (I)

- We are starting to see convergence, but not quite. Try better metrics to improve Markovianity.
- Explore the most stable states
- Explore major relaxation modes



Where to go from here (II)

- Try the other split of the dataset. (1 vs 2 vs 3). Are they independent?
- Use all of them together.
- Try the other systems in, like `villin` and `cxcl12`
 1. Familiarize with the PDB
 2. Devise a reasonable metric

MD analysis libraries

Where to go from here (III)

- R is fine, but *specialized* environments for MD analysis are more productive
 - HTMD[°], MdAnalysis, MdTraj, Bio3D, ... *
 - Most are Python based
- [°] It does:
 - Molecule/trajectory visualization
 - Markov Modeling (etc)

* GitHub: [giorginolab/analysis_libraries_chapter](https://github.com/giorginolab/analysis_libraries_chapter)

*

Analysis libraries for molecular trajectories: a cross-language synopsis

Toni Giorgio

October 1, 2018

To appear in:

Biomolecular Simulations: Methods and Protocols
Edited by M. Bonomi and C. Camilloni

Corresponding author's address/affiliation:

Biophysics Institute, National Research Council of Italy
Department of Biosciences, University of Milan
Via G. Celoria 26, I-20133, Milan, Italy

Running head:

MD analysis libraries: a synopsis

Summary

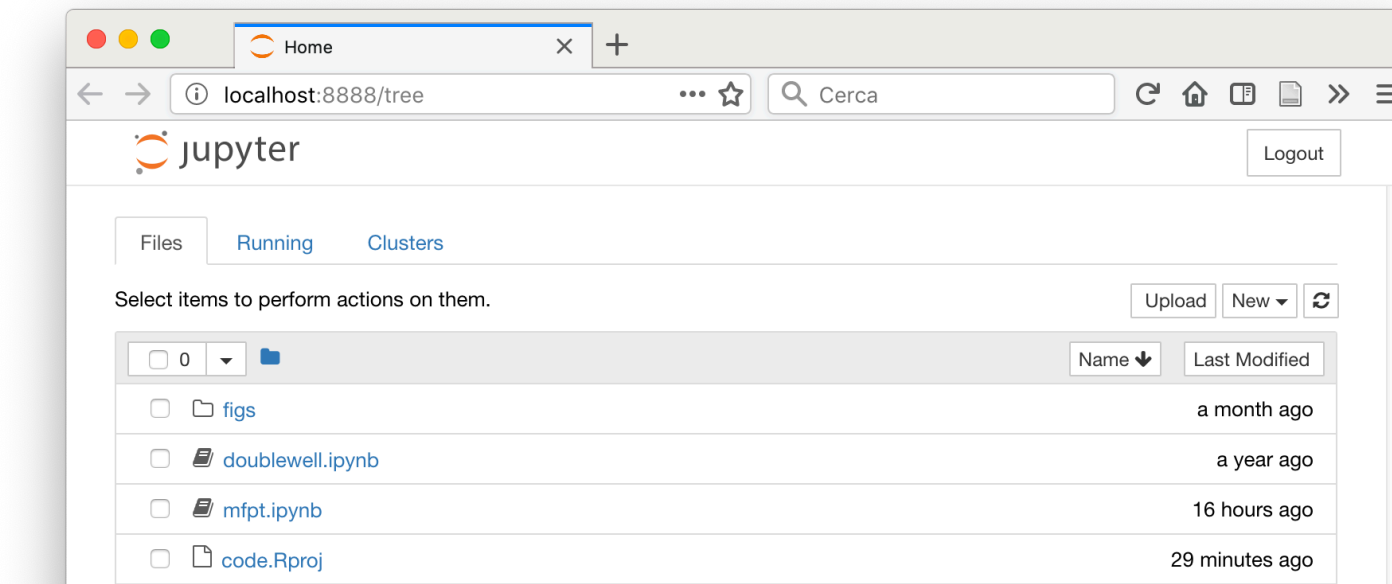
Analyzing the results of molecular dynamics (MD)-based simulations usually entails extensive manipulations of file formats encoding both the topology (e.g. the chemical connectivity) and configurations (the trajectory) of the simulated system. This chapter reviews a number of software libraries developed to facilitate interactive and batch analysis of MD results with scripts written in high-level, interpreted languages. It provides a beginners' introduction to MD analysis presenting a side-by-side comparison of major scripting languages used in MD, and show how to perform common analysis tasks within the VMD, Bio3D, MDTraj, MDAnalysis and HTMD environments.

1 Introduction

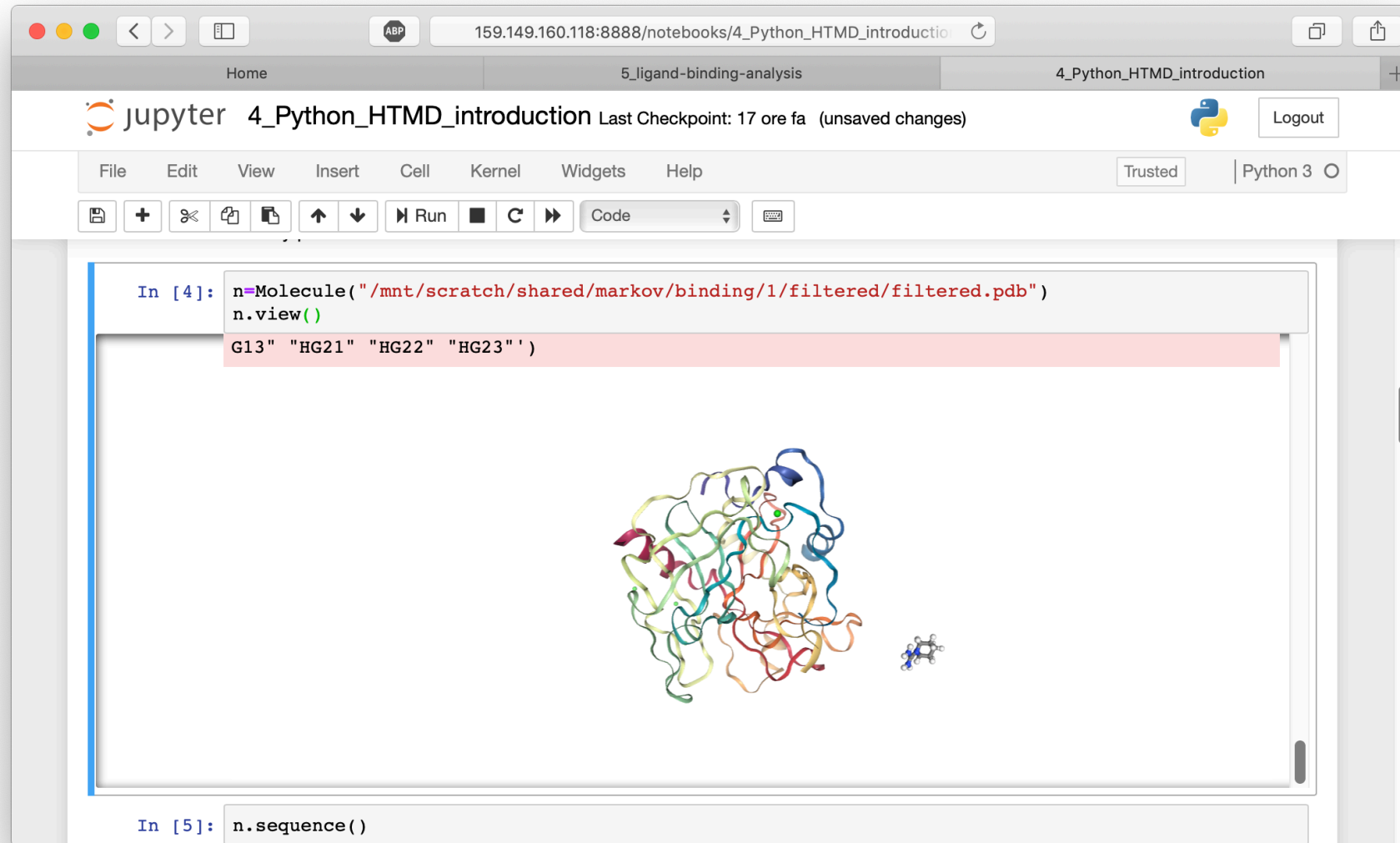
The backbone of molecular dynamics (MD) based methods is to integrate the equations of motion of a system with a given Hamiltonian. The integration is performed by an MD engine with a finite time-step, sufficiently fine to capture

Python Notebooks

- Install the conda environment
- Start with “**jupyter notebook**”
- You will see an interactive Python (*jupyter*) Notebook.



Python Notebooks



The screenshot displays a Jupyter Notebook web interface in a browser window. The address bar shows the URL `159.149.160.118:8888/notebooks/4_Python_HTMD_introduction`. The notebook's title bar indicates the current file is `4_Python_HTMD_introduction`, with tabs for `Home`, `5_ligand-binding-analysis`, and the active file. The Jupyter logo and the text `jupyter 4_Python_HTMD_introduction` are visible, along with a `Last Checkpoint: 17 ore fa (unsaved changes)` status and a `Logout` button. A menu bar includes `File`, `Edit`, `View`, `Insert`, `Cell`, `Kernel`, `Widgets`, and `Help`. A `Trusted` status indicator and a `Python 3` kernel selector are also present. The toolbar contains icons for saving, adding, deleting, and duplicating cells, as well as navigation and execution controls. The main content area shows a code cell with the following Python code:

```
In [4]: n=Molecule("/mnt/scratch/shared/markov/binding/1/filtered/filtered.pdb")
n.view()
G13 "HG21" "HG22" "HG23")
```

Below the code, a 3D ribbon diagram of a protein structure is rendered, colored by residue type. A small molecular model is also visible in the bottom right corner of the visualization area. At the bottom of the interface, a new code cell is partially visible with the text `In [5]: n.sequence()`.

Python Notebooks (MSM)

