Standardization (or not) of File Formats

Erik Lindahl, KTH Royal Institute of Technology & Stockholm University

Partners

Funding

Horizon 2020
European Union Funding
for Research & Innovation

Ten simple rules on

"How to create open access and reproducible molecular simulations of biological systems".

Arne Elofsson, Berk Hess, Erik Lindahl, Shina Caroline Lynn Kamerlin, Alexey Onufriev, David van der Spoel, Anders Wallqvist

Molecular dynamics and other type of simulations have become a fundamental part of life sciences. The simulations are dependent on a number of parameters such as force fields, initial configurations, simulation protocols and software. Researchers have different opinions about the types of software they prefer, and in general we believe authors should be free to choose the tools that best fit their needs. However, as scientists we also have a common obligation to critically test each other's statements to find mistakes (including errors in the algorithms), and we believe PLoS has a particularly strong responsibility to lead this development even if it might cause some short-term grief.

In particular, all published results should, in principle, be possible to reproduce independently by scientists in other labs using different tools. To ensure this, we propose a set of standards that any publication in PLoS Computational Biology should follow.

These rules should not only be limited to molecular dynamics, but also include Monte Carlo simulations, Quantum Mechanics calculations, Molecular docking and any other computational methods involving calculations on biological molecules.

1. Simulation Protocols The authors should provide the complete set of input files that are used in the simulations.
2. Topology and Parameters All topology and parameter files used in the simulations should be provided and made publicly available so they can be implemented and tested with a different program if necessary. That means the files should either be in human-readable format, or the conversion rules should be be publically available.
3. Coordinate Files To ensure maximum reproducibility, the authors should provide the input coordinate files for the simulations in the appropriate formats for the software used. The input files to initiate the the simulations should be provided in a format ensuring that a reader can run the simulations themself. That means the files should either be in human-readable format, or the conversion rules should be be publically available.
4. Software Information Reviewers and readers must be able to reproduce results with as much detail as possible. This means authors need to provide enough details so that the work can be repeated with widely available programs, or that the software is provided. In particular, indicate the specific version of the software package used in the simulation. To further improve reproducibility, we encourage software authors to add information about compilers, flags and the hardware used to log files.
5. Simulation Results Following the PLoS editorial policy for data access, the authors should deposit representative snapshots from the trajectories/simulations in FAIR (Findable, Accessible, Interoperable, Reusable) public repositories, which are dense enough so that the biological insights are supported, and so that new analyses and publications can be performed using the deposited data.

5. Simulation Results Following the PLoS editorial policy for data access, the authors should deposit representative snapshots from the trajectories/simulations in FAIR (Findable, Accessible, Interoperable, Reusable) public repositories, which are dense enough so that the biological insights are supported, and so that new analyses and publications can be performed using the deposited data.
6. Simulation Methods If the software used is not publicly available, the simulation method should be provided or be already published in sufficient details that the results can, in principle, be reproduced using publicly available software.
7. Docking Studies For all studies including screening and docking, the complete set of molecules tested as well as the force fields used and the poses imposed should be publicly available either as databases or detailed descriptions.
8. QM Methods For QM studies, the authors need to provide the following information: absolute energies and energy breakdowns, the level of theory used, the basis set used, the optimization algorithm used, and Cartesian coordinates of all optimized stationary points.
9. Statistical Tests. As in all scientific studies, statistical rigour is necessary in computational studies to evaluate the significance of an observation because molecular dynamics as well as Monte Carlo simulations are stochastic by nature. Appropriate estimates of statistical uncertainty is therefore necessary and should be included for each relevant finding.
10. Be Nice. Remember that we all are a community, so sharing your data, methodologies, software and results in such a way that other can use it will make the entire community thrive. This also applies to the readers - they should not expect unlimited support for using in-house software or methods. Just the fact that it is available provides an important resource for the community.

WHY DO WE EVEN NEED COMMON FILE FORMATS?

- Encourage re-use of existing data to promote science
- Make it trivial for my students to check an old result with a different code
- Facilitate high-throughput usage of multiple programs
- Improve reproducibility
  - Exact coordinates of starting/intermediate/final conformations
  - Exact specifications of algorithms and settings (including workflows)
  - Exact specifications/logs of program settings
- Well-defined & interoperable are important goals - but is “formal standard” one?
- File formats might help (mildly) enforcing users to store important metadata

TRAJECTORY DATA NEEDS

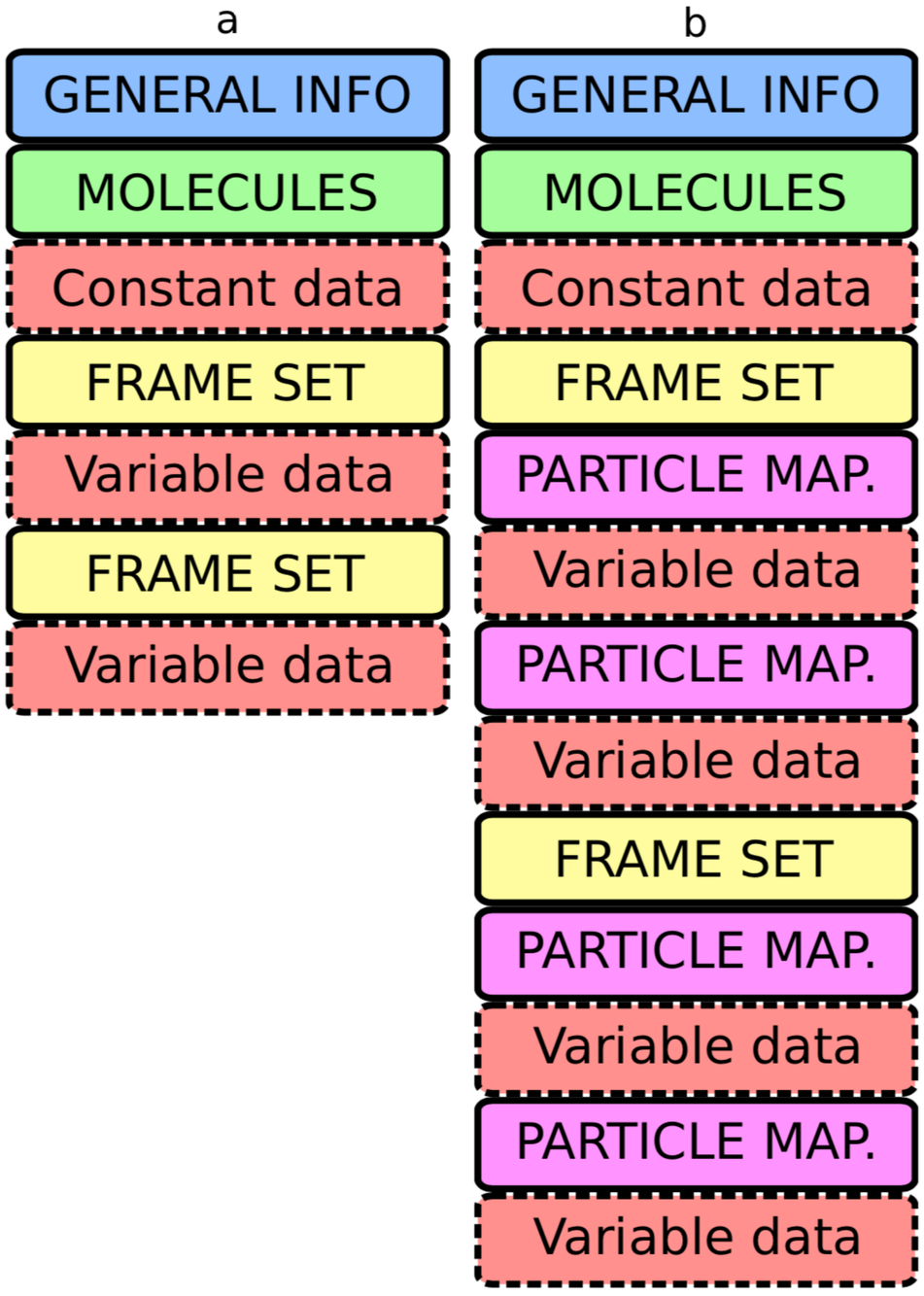
- Provenance records
  - What program/user/run generated the data, and what was the previous step?
- Highly efficient storage
  - Compressed, and allow future even better compression algorithms
  - Lossy & lossless options
  - Efficient parallel IO
  - Hashes & digital signatures for data integrity
- We need both single frame (“conformation”) and trajectory options
  - Balance between convenience and efficiency/space for single frame storage

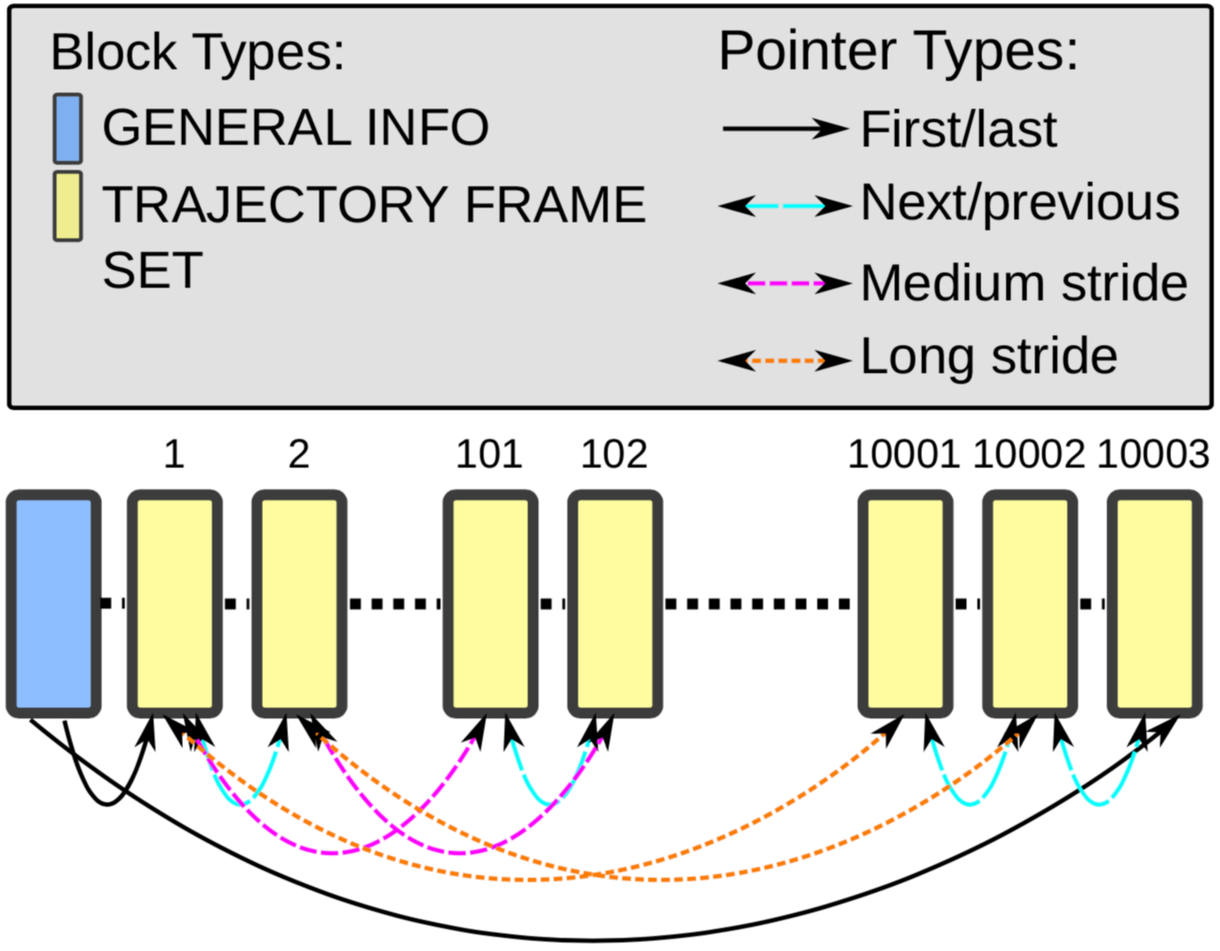
FORCE FIELDS, OPTIONS & METADATA

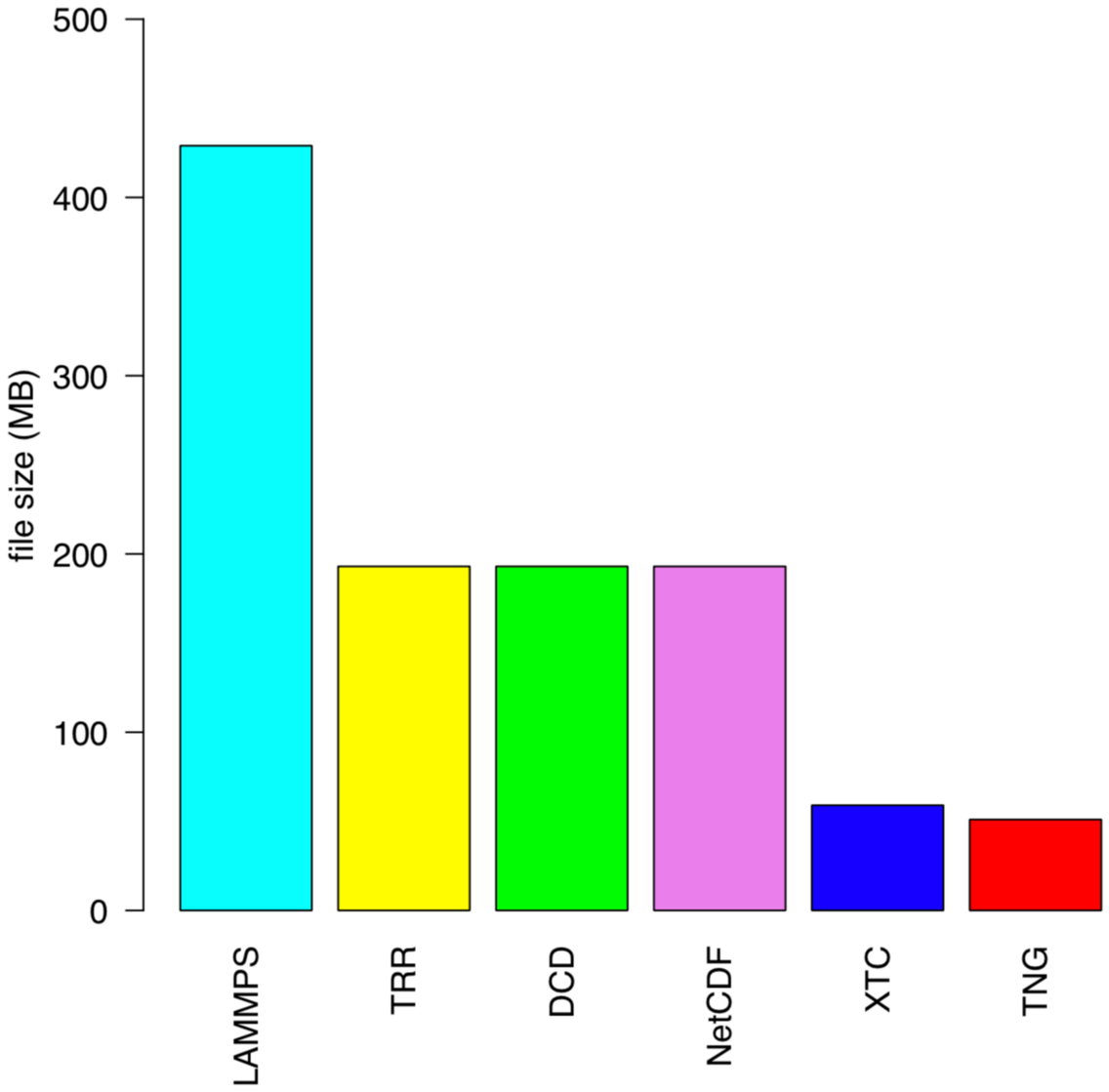
- Verbatim and detailed formats (units, all choices documented, etc.)
- Easy for users to read (and write) without lots of open/close tags
- Make it possible to specify a full simulation/whatever from a single file where options, parameters and coordinates are stored in a single file
- VERY high performance text IO required for this to be realistic
- Extremely portable & completely free (BSD/PD) implementations required
- Balance:
  - Core set of truly common options that we all agree on
  - Need a path for group(s) to expand with their own features/algorithms

TRAJECTORY DATA THOUGHTS

- Think MKV / QuickTime / AVI
- Containers with high-level properties & APIs - low-level formats that can evolve
- Needs C, C++, Fortran and Python APIs
- Endian, precision, and IEEE-format portability
- Enable temporal/multi-frame compress (I/B frames)
- Fast (instant) random access to frames
- Focus on “state”, i.e. properties that cannot be re-computed from trajectory
- Some examples from a semi-recent GROMACS project: TNG


a
GENERAL INFO
MOLECULES
Constant data
FRAME SET
Variable data
FRAME SET
Variable data
b
GENERAL INFO
MOLECULES
Constant data
FRAME SET
PARTICLE MAP.
Variable data
PARTICLE MAP.
Variable data
FRAME SET
PARTICLE MAP.
Variable data
PARTICLE MAP.
Variable data

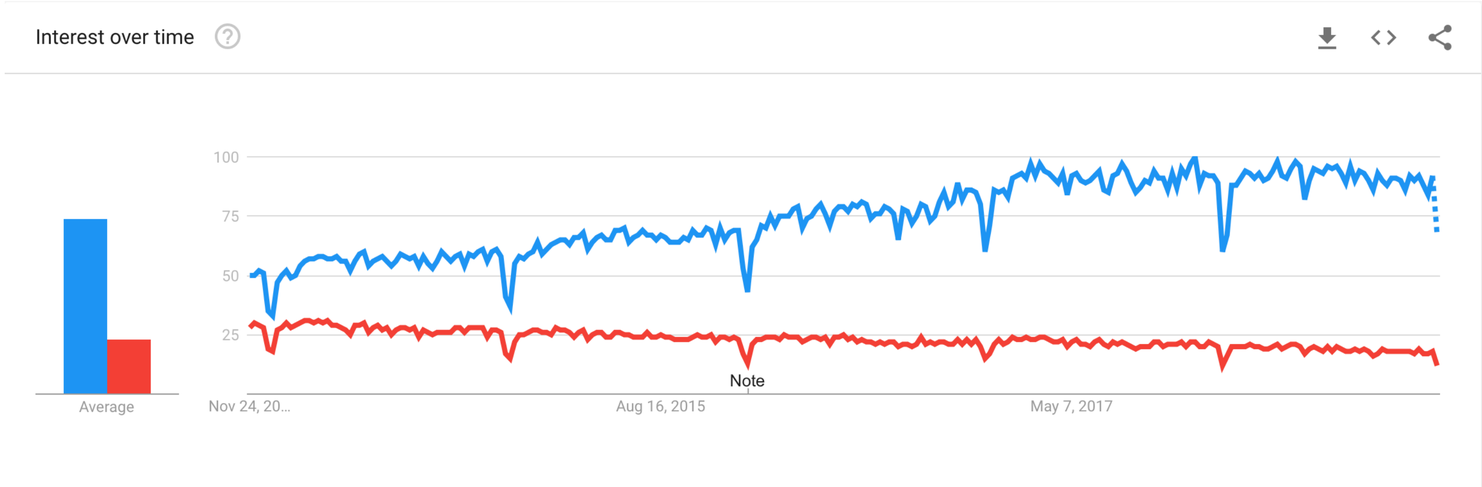

Block Types:
GENERAL INFO
TRAJECTORY FRAME SET
Pointer Types:
First/last
Next/previous
Medium stride
Long stride
1
2
101
102
10001
10002
10003



Software	**Version**	**Size (MB)**
AMBER	11	4
GROMACS	4.6.2	11
NAMD	2.8	8
NetCDF		5
HDF5		8
TNG	1.4	0.2

PARAMETER DATA THOUGHTS

- Stick to simple Key-Value Trees
  - Formats evolve, but this should be easy to implement in any new format
- Proposed current language to focus our efforts on: JSON (subset of YAML)
- Meta-metadata for everything would be great (list of molecules in simulation?)
  - … but starting with *something* is more important
- Reasonable goal: Have single JSON file contain entire simulation starting state?
- How do we handle an ensemble of 10,000 runs?



JSON

XML

XML

```
<empinfo>
    <employees>
        <employee>
            <name>James Kirk</name>
            <age>40</age>
        </employee>
        <employee>
            <name>Jean-Luc Picard</name>
            <age>45</age>
        </employee>
        <employee>
            <name>Wesley Crusher</name>
            <age>27</age>
        </employee>
    </employees>
</empinfo>
```

JSON

```
{   "empinfo"  :
        {
                "employees" :   [
                {
                        "name" : "James Kirk",
                        "age" : 40,
                },
                {
                        "name" : "Jean-Luc Picard",
                        "age" : 45,
                },
                {
                        "name" : "Wesley Crusher",
                        "age" : 27,
                }
                                        ]
        }
}
```

THE SHARING STRATEGY

- File formats is only a beginning
  - Arguably more important: We need to have a clear high-level standard for how *sharing* itself happens, rather than worrying about the lowest-level formats
  - Can we design a standard for commands/specifications that allow me to archive my data, and for me to access data for https://doi.org/10.1109/5.771073?
  - If we “control” and work through the large codes, research groups and MolSSI/BioExcel, we might be able to push this in the community as a de-facto standard
  - Requirements:
    - Extremely resilient, distributed
    - Initial store only metadata, and track the raw trajectories stored elsewhere

Search Torrents | Browse Torrents | Recent Torrents | TV shows | Music | Top 100

Pirate Search

Preferences
Languages

All Audio Video Applications Games Porn Other

Pirate Search | I'm Feeling Lucky

How do I download?

Login | Register | Language / Select language | About | Blog
Usage policy | TOR | Doodles | Forum

BTC: **3HcEB6bi4TFPdvk31Pwz77DwAzfAZz2fMn**
BTC (Bech32): **bc1q9x30z7rz52c97jwc2j79w76y7l3ny54nlvd4ew**
LTC: **LS78aoGtfuGCZ777x3Hmr6tcoW3WaYynx9**
XMR: **46E5ekYrZd5UCcmNuYEX24FRjWVMgZ1ob79cRViyfvLFZjfyMhPDvbuCe54FqLQvVCgRKP4UUMMW5fy3ZhVQhD1JLLufBtu**

SHOULD WE THINK OF OUR (LARGE OBJECT) DATA SHARING IN THESE TERMS INSTEAD?

- “Torrent” metadata file describing the object
- Download from multiple sources
- The more popular data is, the more available it will be (and with higher bandwidth)
- When you have downloaded a while, you would help “seed” the object
- Multi-level approach that might be easier to achieve:
  - Initially, we would only need the equivalent of a metadata tracker (EBI, PDB?)
  - Small groups can store their data on Zenodo/OSF/Google/Amazon
  - Larger groups can use their local center or their own storage
  - The group’s own data repository could also be the sharing repository
  - Ping-back to track the current number of copies of the data world-wide

SOME THOUGHTS TO KICK OFF THE DISCUSSION

- No community standard ever emerged from a single group/lab
- We need buy-in from several labs, representing many different codes
- Doing it together is a natural way of kick-starting a pseudo-standard
- What can we concretely contribute (BSD/PD codebases)?
- What would we be willing to help implement?
- Maybe less noble, but important to keep it going longer term:
  What direct scientific impact do we hope to achieve? Funding? Student work?
- Where / how can be compromise to make this feasible?

- **Don’t repeat the mistake of coming up with ideas for perfect file formats at this meeting, and then it is too complex for anybody to implement quickly afterwards…**