Ensemble simulations with experimental restraints: Running at Scale
-or-
a designed interface for MD simulations

Peter Kasson, University of Virginia

\[ U = \frac{1}{2} K (P_{MD}(x) - P_{DEER}(x))^2 \]
Motivation:

• Most molecular dynamics simulation packages are monolithic

• Most applications are not

• How can we enable flexible, performative use of large-scale MD simulations?
Motivation II

- Custom modifications of MD code are common, e.g. to implement additional potentials for experimental data

- Having open code that can be modified is great

- Having everyone modify it is a Bad Idea for barrier-to-entry, performance, reproducibility, maintainability

- Goal: custom modifications with no changes to core MD code.
Motivation III

- Not just another Python API
- Design inspiration from TensorFlow (API compatible)
- Procedural commands set up a computational graph
- Enables user-transparent performance optimizations

![Diagram](image:deepideast.net)
Motivation IV

- Ensembles as first-class entities, computational context abstracted

- Description of what to execute should be distinct from specification of where to execute.

![Diagram showing the flow of processes involving input structures, run parameters, simulation setup, MD simulations, trajectory analysis, ensemble results, and further iterations.](image)
gmxapi: Python (and more) API for Gromacs

- Running molecular dynamics simulations
- Clean interface that permits scalable, optimized execution with standard or custom code
- Build and execute a computational graph
What does this mean in practice?

```python
import gmx
import myplugin

md = gmx.workflow.from_tpr(tpr_list)
potential = gmx.workflow.WorkElement(
    namespace="myplugin",
    operation="ensemble_restraint",
    depends=[],
    params=params)
potential.name = "ensemble_restraint_1"
md.add_dependency(potential)

context = gmx.context.ParallelArrayContext(md)
with context as session:
    session.run()
```

Simple, clean code.

This runs an ensemble of simulations using a custom potential.
A peek under the hood

**a**

```python
>>> md = gmx.from_file([filename1, filename2, filename3, ...])
```

**b**

```python
>>> potential = myplugin.EnsembleRestraint(sites, *args, **kwargs)
>>> md.add_dependancy(potential)
```

**c**

```python
>>> gmx.run()
```
Custom potentials are easy(ish)

- Boilerplate C++ code, en route to templated classes
- essentially implement a calculate() method and then an update callback to update potentials for adaptive approaches.
What this enables

• Implementation of restrained-ensemble simulation (Roux, 2013)

• Simple, scalable execution within 5% of native code performance.

• Can either execute multiple iterations with independent ensemble members or update potentials with a (blocking) reduce
Example of restrained-ensemble simulation

Ensemble of 20 simulations
ensemble reduce & potential update every 100 ps

Update pull potential:

\[
U = \frac{1}{2} K (P_{\text{MD}}(x) - P_{\text{DEER}}(x))^2
\]

Hays et al., 2018
Example of restrained-ensemble simulation and convergence

Convergence to experimental distributions
(Assessed via histograms, Jensen-Shannon divergence)

Hays et al., 2018
Current status

• Functionality: Can run ensembles, enable custom force plugins, perform ensemble reduce as needed.

• Implementations: Two different ensemble-biasing algorithms implemented, working on automated umbrella sampling next.

• Alpha/beta code release [github.com/kassonlab/gmxapi](https://github.com/kassonlab/gmxapi)

• (almost) running on Blue Waters

• Next round of functionality in progress. Taking requests.
We want to hear from you!

- gmxapi manuscript: Irrgang et al., 2018 doi:10.1101/306043

- Restrained ensembles: Hays et al., 2018 doi: 10.1101/319335

- Github: https://github.com/kassonlab/gmxapi

- Want to join as a scientist or developer? We’re looking for students and postdocs!

- kasson@virginia.edu
Acknowledgements

Eric Irrgang
Jennifer Hays
Cara Broshkevitch
Ricardo Ferreira
Isabel Goronzy
Ania Pabis
Kenta Okamoto
Ana Villamil Giraldo

Collaborators:
Mark Abraham
Linda Columbus
Shantenu Jha
Erik Lindahl
Michael Shirts

Funding:
(for this work)
NIGMS R01GM115790
BW GLCPC and Graduate Fellowships (Kasson, Hays)
MolSSSI Fellowships