



University of
Nottingham

UK | CHINA | MALAYSIA

Ab initio molecular dynamics and machine learned potentials based molecular dynamics

Lei Lei: Lei.Lei2@nottingham.ac.uk

Sanliang Ling: Sanliang.Ling@nottingham.ac.uk

University of Nottingham



Outline

- Introduction to molecular dynamics
- *Ab initio* molecular dynamics (AIMD) using CP2K
- Hands on AIMD simulation and data analysis on Archer2
- Neural network potentials (NNP) as force evaluation method
- Hands on MD simulation using NNP



Goals of Molecular Dynamics simulations and data analysis

- MD simulations: obtain trajectory data by solving equations of motion

$$F_i = m_i \ddot{r}_i$$

Newton

$$\mathcal{H}(r, p) = \frac{|p|^2}{2m} + \mathcal{U}(r)$$

Hamiltonian

- MD data analysis: calculate ensemble average of properties from trajectory data

$$\langle A \rangle = \int P(p, t) A(p, r) dp dr = \int A(p(t), r(t)) dt$$

A : target property, P : probability, p : momenta, r : position



Ensembles in MD

Ensemble: all microstates (r, p) accessible to simulation, each microstate occurring with a particular probability

Various possibilities for quantities that may be conserved or fixed in the simulation include:

N: particle number

V: volume

P: pressure

E: energy

T: temperature

μ : chemical potential (not implemented in CP2K)

- NVE: microcanonical

- NVT: canonical

- NPT: isothermal-isobaric



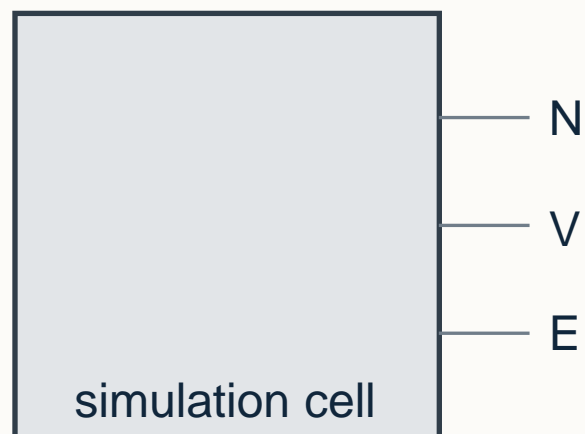
Ensembles in MD: NVE

Isolated system with constant number of particles N , volume V and energy E

Total energy is conserved as the system is isolated

Solving equations of motion without temperature or pressure control

$$F_i = m_i \ddot{r}_i$$



- Drift in E due to rounding and truncation errors
- Dynamical variables well defined
- Required initial conditions: positions and velocities

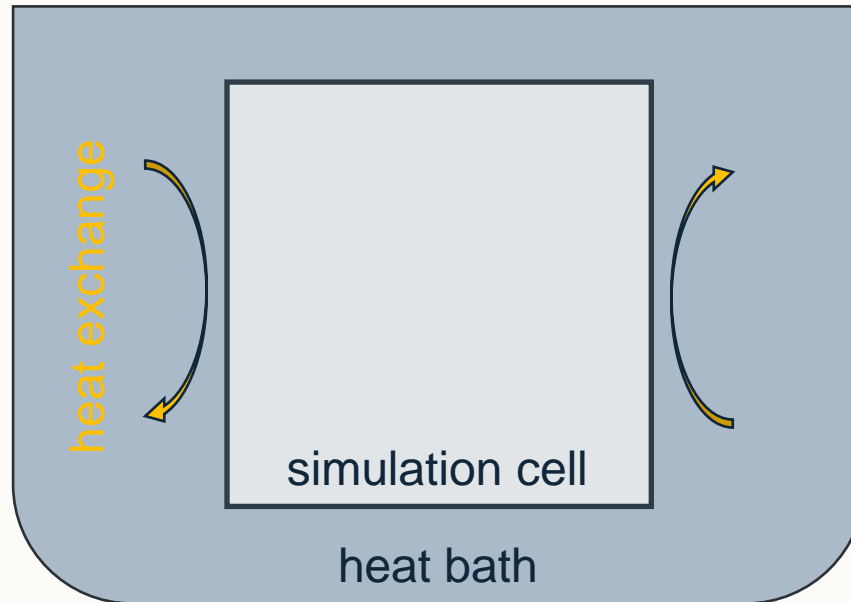


Ensembles in MD: NVT

System with constant number of particles N , volume V and temperature T (controlled by a thermostat)

Total energy is no longer conserved

- Energy exchange between system and heat bath exists
- Constant of motion: energy of the system + heat bath



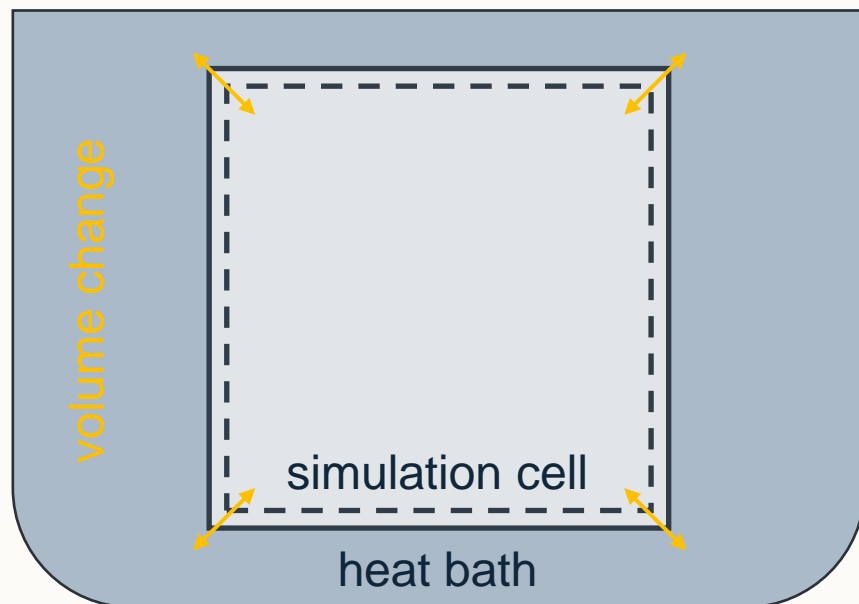


Ensembles in MD: NPT

System with constant number of particles N , pressure P (controlled by a barostat) and temperature T (controlled by a thermostat)

Box size/shape is allowed to change in response to internal stress and external pressure

Energy exchange with the environment: $dW = PdV$



Constant of motion =
energy of the system +
energy of the thermostat +
energy of the barostat



Ensembles in CP2K

```
&MOTION
```

```
&MD
```

```
ENSEMBLE NVE
```

```
STEPS 1000
```

```
TIMESTEP 0.5
```

```
TEMPERATURE 300.0
```

```
&END MD
```

```
&END MOTION
```

Choices:

- NVE: microcanonical
- NVT: canonical
- NPT_F: isobaric-isothermic
- NPT_I: isobaric-isothermic in isotropic cell
- NPE_F: constant pressure
- NPE_I: constant pressure in isotropic cell
- LANGEVIN: canonical using Langevin dynamics
- ISOKIN: constant kinetic energy
- HYDROSTATICSHOCK, MSST, MSST_DAMPED, NVT_ADIABATIC ...



Thermostats: velocity rescaling

Obtain desired temperature T_0 by a factor:

$$\lambda = \sqrt{\frac{T_0}{T(t)}}$$
$$\Delta T = \frac{1}{2} \sum_{i=1}^N \frac{2}{3} \frac{m_i (\lambda v_i)^2}{N k_B} - \frac{1}{2} \sum_{i=1}^N \frac{2}{3} \frac{m_i v_i^2}{N k_B} = (\lambda^2 - 1) T(t)$$

Straight forward

Does not correspond to any ensemble



Thermostats in CP2K: LANGEVIN

Add a dissipative frictional force and a stochastic force:

$$m\ddot{r}_i = -\frac{\partial U}{\partial r_i} - m\Gamma\dot{r}_i + W_i(t)$$

Γ : friction coefficient
 W : random force

Relation between magnitude of force and friction:

$$\langle W_i(t), W_j(t') \rangle = \partial_{ij} \delta(t - t') 6m\Gamma k_B T$$

Magnitude of the perturbation depends on the instantaneous temperature

Surprisingly useful in practice!



Thermostats in CP2K: LANGEVIN

Produces canonical ensemble (NVT)

Local thermostat

Ergodic

Allows the use of large time steps

But:

- does not conserve momentum (due to drag force)
- only useful for sampling, **cannot be used to calculate transport properties, e.g., diffusion coefficient**



Thermostats in CP2K: Nosé-Hoover (chains)

Add two additional degrees of freedom:

s – position of imaginary heat reservoir

p_s – conjugate momentum of imaginary heat reservoir

Additional parameter

Q – effective mass

$$p_s = \frac{\partial \mathcal{L}}{\partial \dot{s}} = Q \dot{s}$$

Momenta conjugate to \mathbf{r}_i :

$$\mathbf{p}_i = \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{r}}_i} = m_i s^2 \dot{\mathbf{r}}_i$$

Hamiltonian:

$$H_N = \sum_i \frac{\mathbf{p}_i^2}{2m_i s^2} + \mathcal{U}(\mathbf{r}^N) + \frac{p_s^2}{2Q} + g k_B T \ln s$$



Thermostats in CP2K: Nosé-Hoover (chains)

```
&MOTION
&MD
&THERMOSTAT
TYPE NOSE
&NOSE
TIMECON 1000
&END NOSE
&END THERMOSTAT
&END MD
&END MOTION
```

- Produces canonical ensemble (NVT)
- Local thermostat
- Ergodic (Nosé-Hoover chain only)
- Second order – temperature may oscillate around target



Thermostats in CP2K: CSV

Kinetic energy:

$$dK = \sum_i \frac{f_i \cdot p_i}{m_i} dt + (\bar{K} - K) \frac{dt}{\tau} + 2 \sqrt{\frac{K \bar{K}}{N_f}} \frac{dW}{\sqrt{\tau}} \left(\tau = \frac{1}{2\gamma} \right)$$

Thermostat part:

$$dK = \boxed{(\bar{K} - K) \frac{dt}{\tau}} + \boxed{2 \sqrt{\frac{K \bar{K}}{N_f}} \frac{dW}{\sqrt{\tau}}}$$

Berendsen
thermostat

Noise gives correct
fluctuations



Thermostats in CP2K: CSV

```
&MOTION
&MD
&THERMOSTAT
TYPE CSV
REGION GLOBAL
&CSV
TIMECON 50
&END CSV
&END THERMOSTAT
&END MD
&END MOTION
```

- Stochastic velocity rescaling for $\tau = 0$
- Correct fluctuations
- Reserves dynamic properties
- Recovers Langevin for single degree of freedom



Thermostats in CP2K: tips

- Use a small `TIMECON` for rapid equilibration
- Default `TIMECON` is usually OK for production MD
- Check `PROJECT-1.ener` file, ensure that the constant of motion is conserved
- Check large fluctuations in temperature
- Almost all of the same options apply for barostats
 - `MOTION%MD%BAROSTAT`



Result files from a CP2K MD simulation

| | | |
|-------------------|---|-------------------------|
| PROJECT-1.cell | → | Cell parameters |
| PROJECT-1.ener | → | Restart input script |
| PROJECT-1.restart | → | Restart input script |
| PROJECT-1.stress | → | Stress tensor |
| PROJECT-frc-1.xyz | → | Forces on each atom |
| PROJECT-pos-1.xyz | → | Atomic positions |
| PROJECT-vel-1.xyz | → | Velocities of each atom |



Output settings in CP2K: PRINT

```
&PRINT
&TRAJECTORY SILENT
FORMAT  XMOL
&EACH
MD  10
&END EACH
&END TRAJECTORY
...
&END PRINT
```

set write
frequency

```
&PRINT
...
&CELL  SILENT
&EACH
MD  10
&END EACH
&END CELL
...
&END PRINT
```

IR, NMR spectra calculation
input settings: refer to
[spectroscopy tutorial](#)

Write cell parameters when
using NPT ensemble



Force evaluation in CP2K: DFT

```
&FORCE_EVAL  
METHOD    QS  
STRESS_TENSOR    ANALYTICAL  
&DFT  
...  
&END    DFT  
&END    FORCE_EVAL
```

Use Quickstep to calculate force
Include this if using NPT

DFT settings here according to
purpose

Check manual for options and explanations: [CP2K manual](#)



Check energy conservation

Plot `PROJECT.ener` file

- use Jupyter Notebook
- use gnuplot

If the constant of motion is not well conserved, try to:

- make `EPS_SCF` tighter (to reduce drift)
- make `TIMESTEP` shorter (to reduce fluctuations)
- play with `EXTRAPOLATION_ORDER` (to reduce drift and or instabilities)



Restart MD simulation in CP2K

Backup original input script:

```
mv PROJECT.inp PROJECT.inp.bak
```

Make a new directory:

```
mkdir restart
```

Copy the input script and make changes on ensemble, steps etc and add:

```
&EXT_RESTART  
  RESTART_FILE_NAME PROJECT-1.restart  
&END EXT_RESTART
```

Re-submit job

```
sbatch job.slurm
```

Alternatively, copy PROJECT-1.restart and modify as a new input script.



Data analysis of MD simulation

Trajectory visualisation: [VMD](#), [nglview](#)

Transport property, eg. diffusion coefficient : [MDtraj](#), [ASE](#)

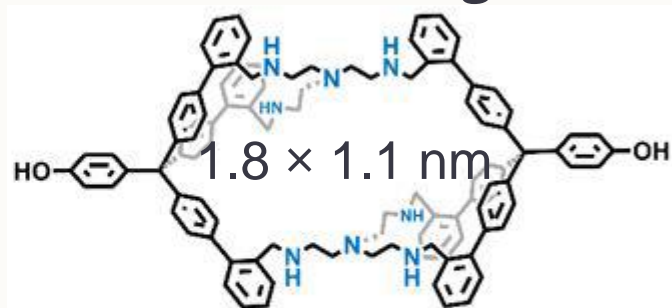
Ensemble average of properties $g(r)$ etc: [MDtraj](#), [TRAVIS](#)

Vibration spectra: [TRAVIS](#)



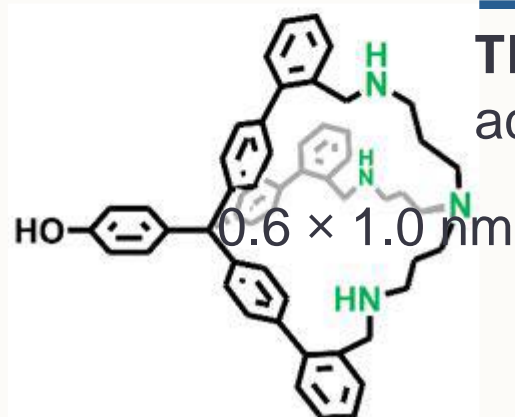
Example: water diffusion in supramolecular cages

Molecular Cage 1



React with

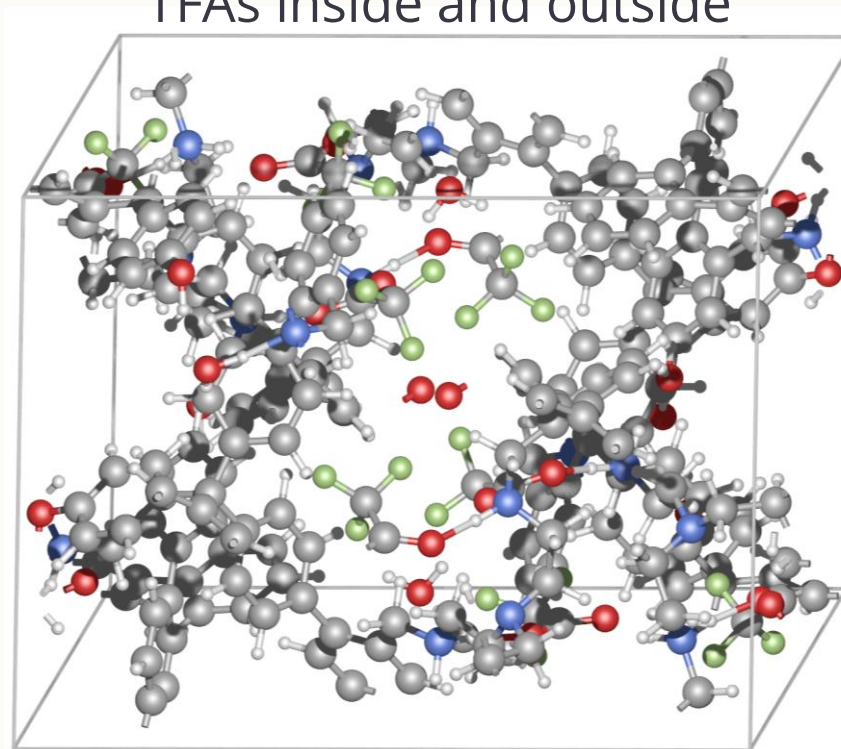
TFA: trifluoroacetic acid



Molecular Cage 2

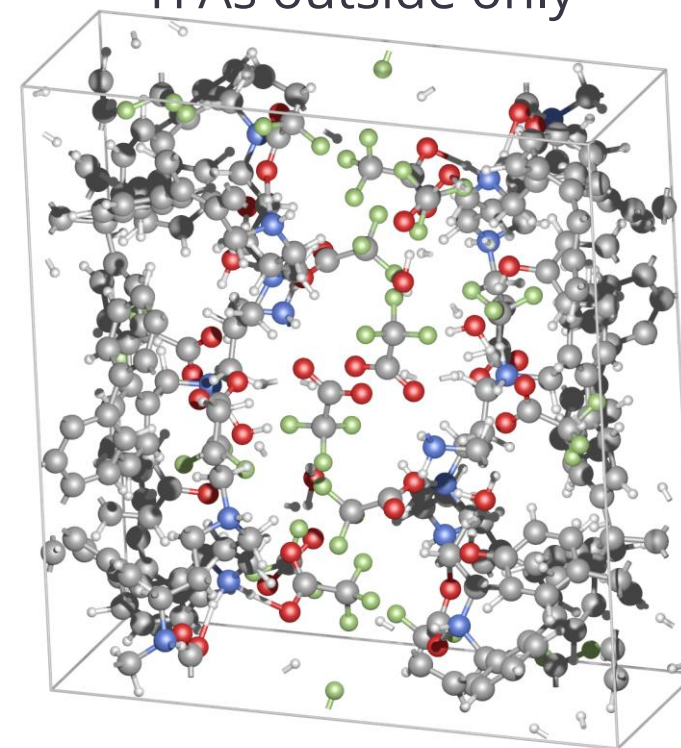
Cage 1

“Plum pudding”
TFAs inside and outside



Cage 2

“Sandwich”
TFAs outside only





My input settings

- Orbital transformation (OT) method
- Full single inverse considering the system size
- VDW

```
&MD
ENSEMBLE NVT
TIMESTEP 0.5
...
&THERMOSTAT
TYPE CSVR
&CSVR
TIMECON 50
&END CSVR
&END THERMOSTAT
&END MD
```

```
METHOD QS
...
&SCF
&OT T
MINIMIZER DIIS
PRECONDITIONER
FULL_SINGLE_INVERSE
&END OT
...
&XC
...
&XC_FUNCTIONAL NO_SHORTCUT
&PBE T
&END PBE
&END XC_FUNCTIONAL
```

```
&VDW_POTENTIAL
POTENTIAL_TYPE
PAIR_POTENTIAL
&PAIR_POTENTIAL
TYPE DFTD3
PARAMETER_FILE_NAME
dftd3.dat
REFERENCE_FUNCTIONAL PBE
CALCULATE_C9_TERM T
&END PAIR_POTENTIAL
&END VDW_POTENTIAL
&END XC
```




My data analysis

Python code FishMol + ASE

Properties:

- Diffusion coefficient using original trajectory
- H bond analysis using wrapped trajectory
 - H bond recognition
 - Radial distribution function, angular distribution function, combined distribution function
 - H bond lifetime analysis based on time auto correlation function

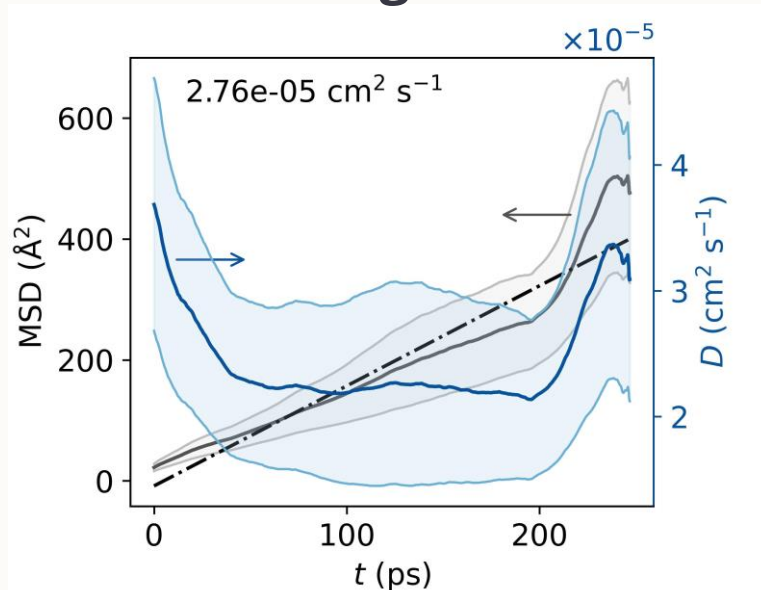


Example: water diffusion in supramolecular cages

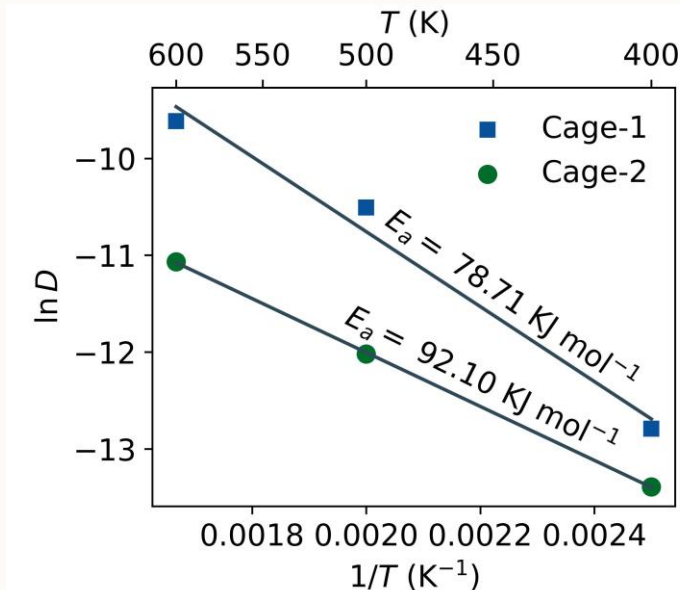
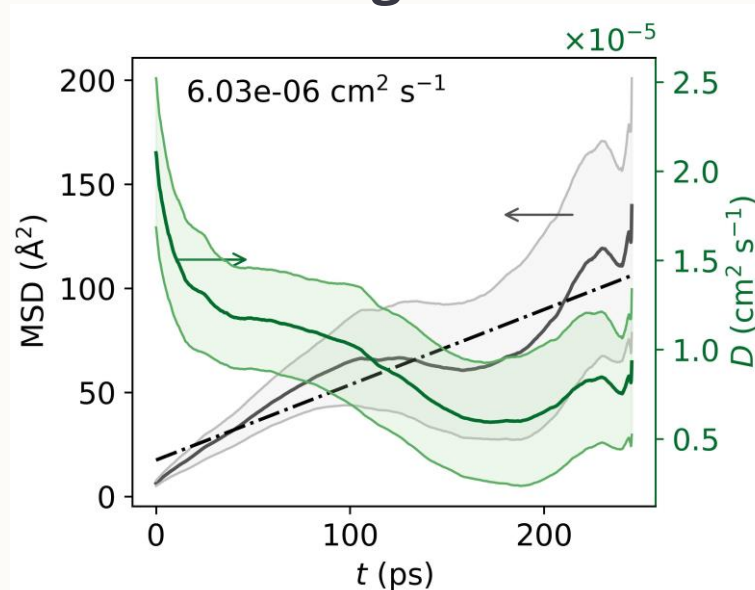
MSD: mean square displacement

D : water diffusion coefficient

Cage 1



Cage 2



Simulation: D of Cage 1 is **4.5** times larger

Experimental data: Cage 1 is **100** times more proton conductive

E_a : apparent activation energy



Example: water diffusion in supramolecular cages

Diffusion anisotropy

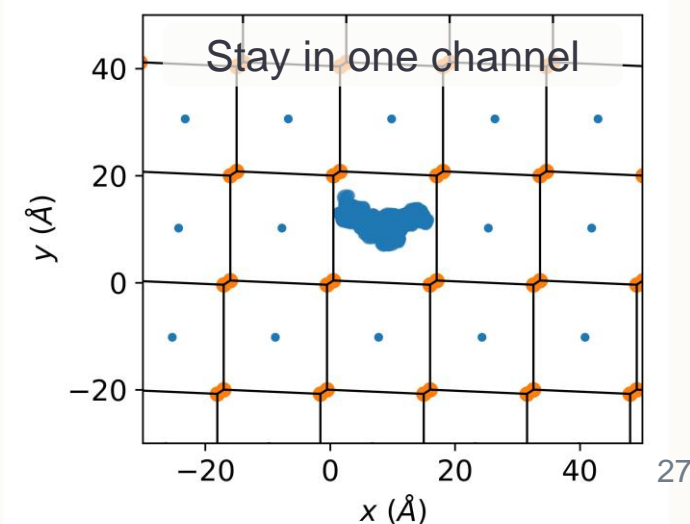
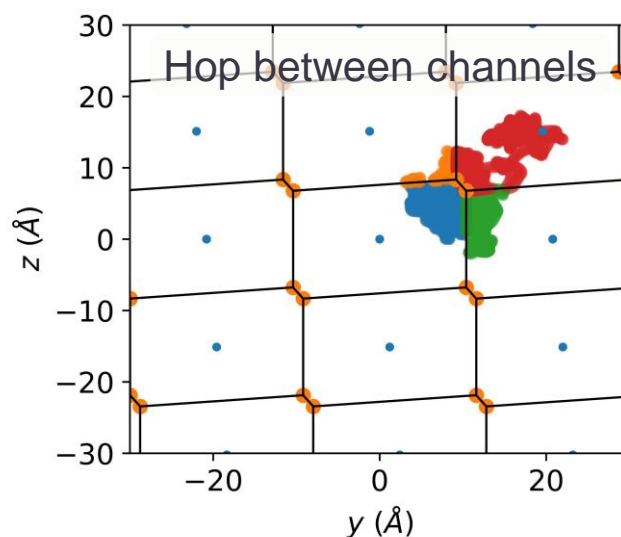
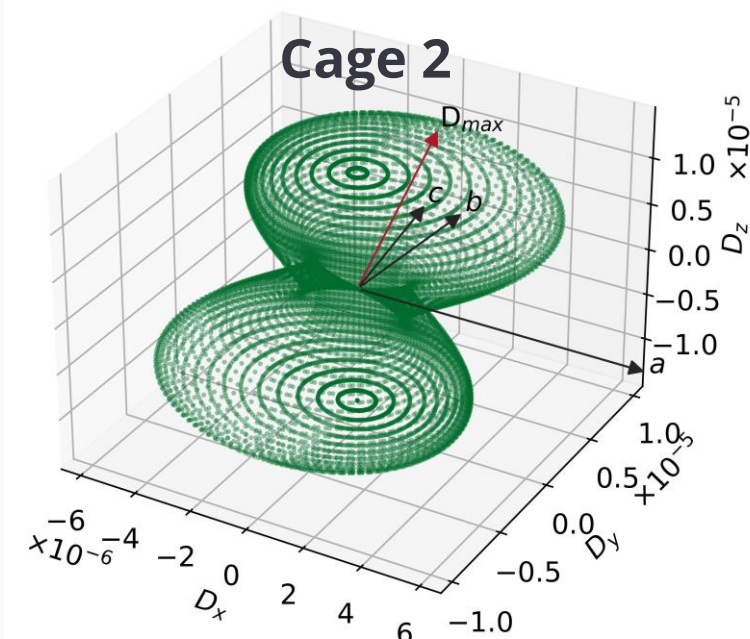
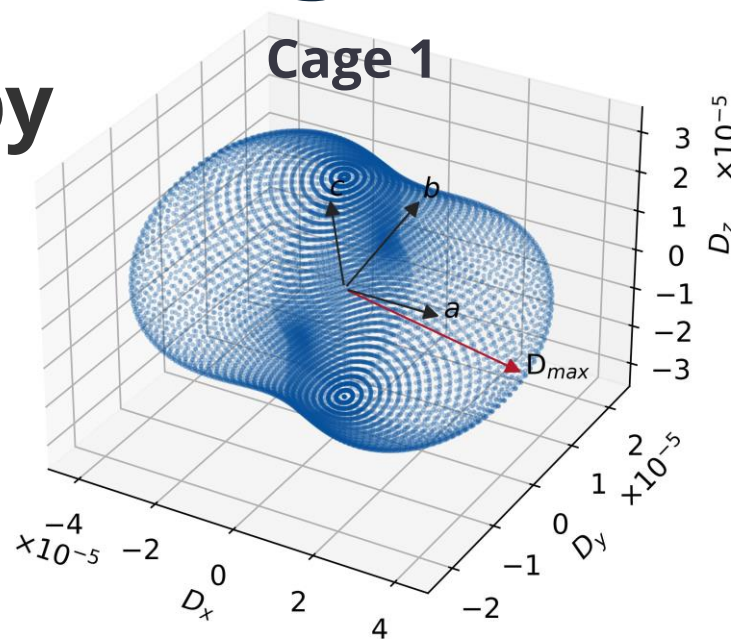
1D diffusion coefficients

Cage 2 has a much smaller “neck”

Voronoi diagrams of diffusion channels

Cage 1: 7/8 switches channels
Cage 2: all stay inside channel

Polycrystalline samples:
high density of grain boundaries





Example: water diffusion in supramolecular cages

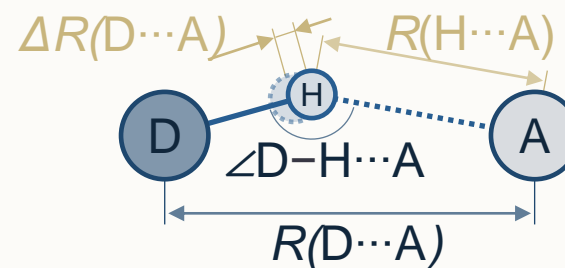
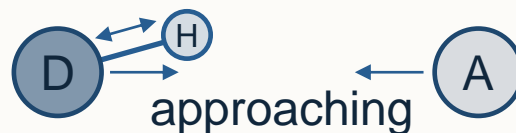
Hydrogen bonding from geometry criteria

Formation of H bond

D: donor atom

A: acceptor atom

lengthening



| Geometry criteria | Strong | Moderate | Weak |
|--|-------------------------|-------------------|---------------------|
| D...A separation: $R(D\cdots A)$ (Å) | 2.2 - 2.5 | 2.5 - 3.2 | > 3.2 |
| D-H...A angle: $\angle D-H\cdots A$ (°) | 170 - 180 | > 130 | > 90 |
| Lengthening of D-H: $\Delta R(D-H)$ (Å) | 0.08 - 0.25 | 0.02 - 0.08 | < 0.02 |
| Proton acceptor distance: $R(H\cdots A)$ (Å) | 1.2 - 1.5 | 1.5 - 2.2 | > 2.2 |
| D-H vs H...A | $D-H \approx H\cdots A$ | $D-H < H\cdots A$ | $D-H \ll H\cdots A$ |

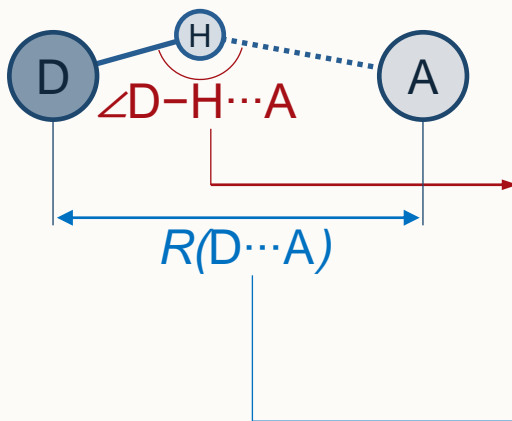


Example: water diffusion in supramolecular cages

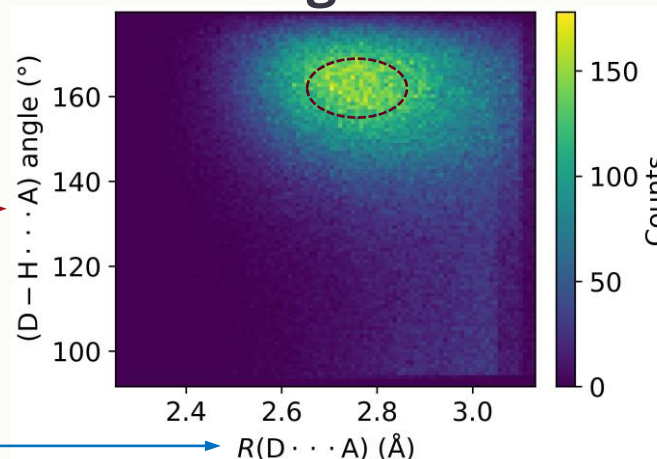
H bond analysis

Strength:

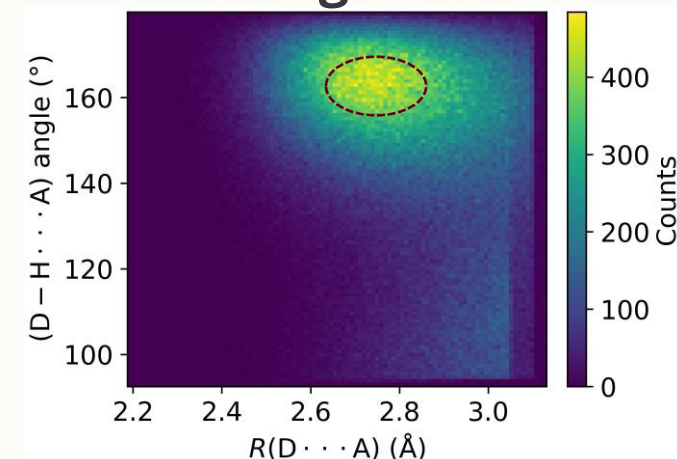
- about the same
- moderate strength (16.74 - 62.76 kJ mol⁻¹)



Cage 1



Cage 2

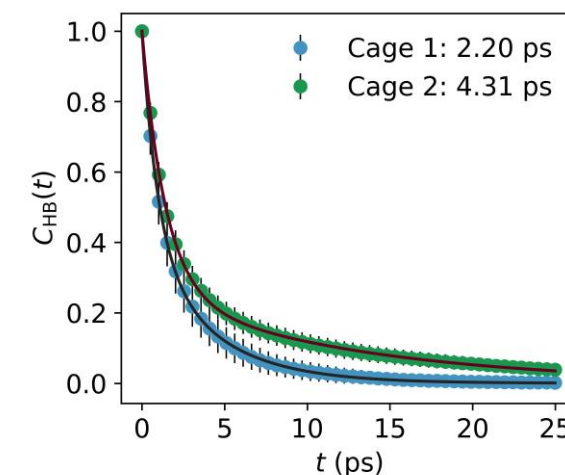
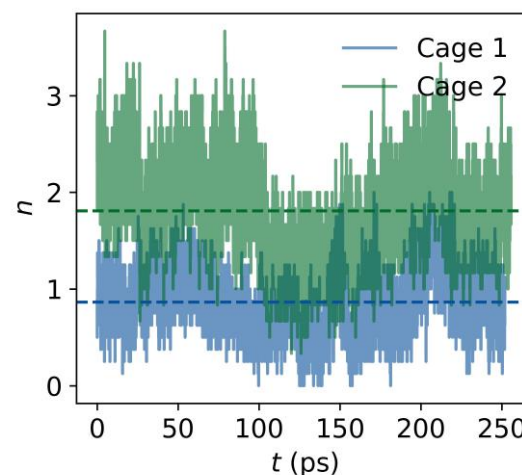


H bonds on water:

One less in Cage 1

H bond lifetime:

Cage 1 $\approx \frac{1}{2}$ Cage 2

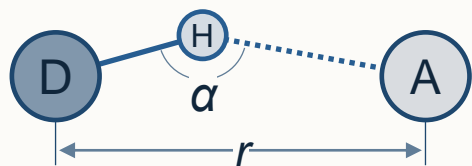




Example: water diffusion in supramolecular cages

H bond behaviours

Strong H bonds:
distinct centres

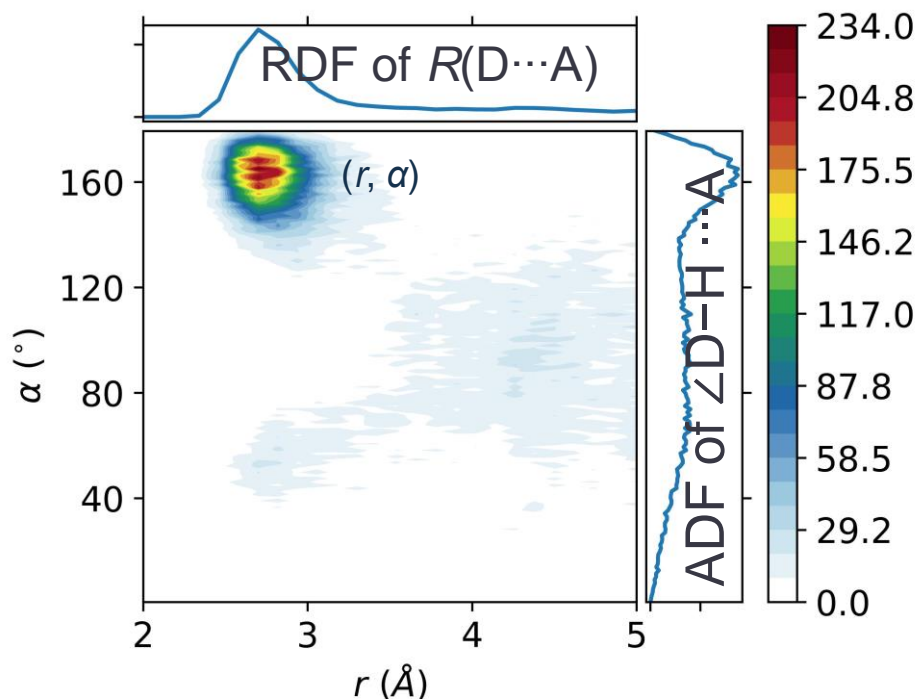


RDF: radial distribution function

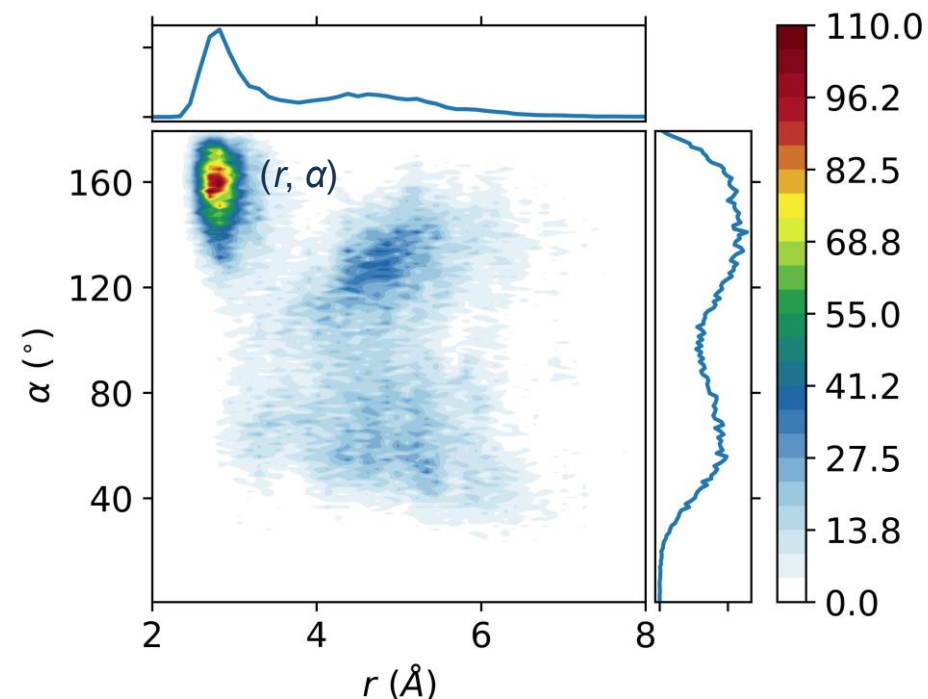
ADF: angular distribution function

CDF: combined distribution function

Donor: water -OH
Acceptor: amine N



Donor: -ph-OH
Acceptor: water O

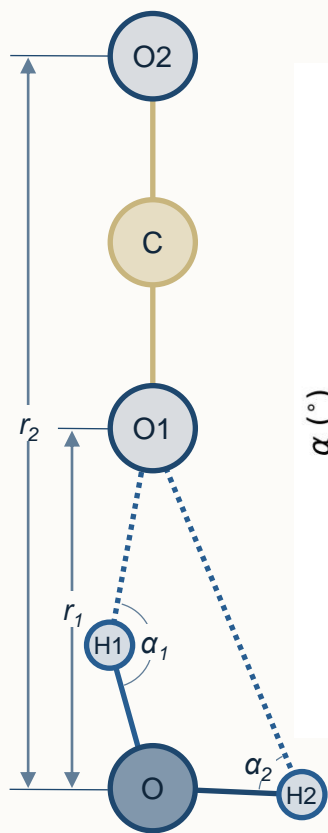




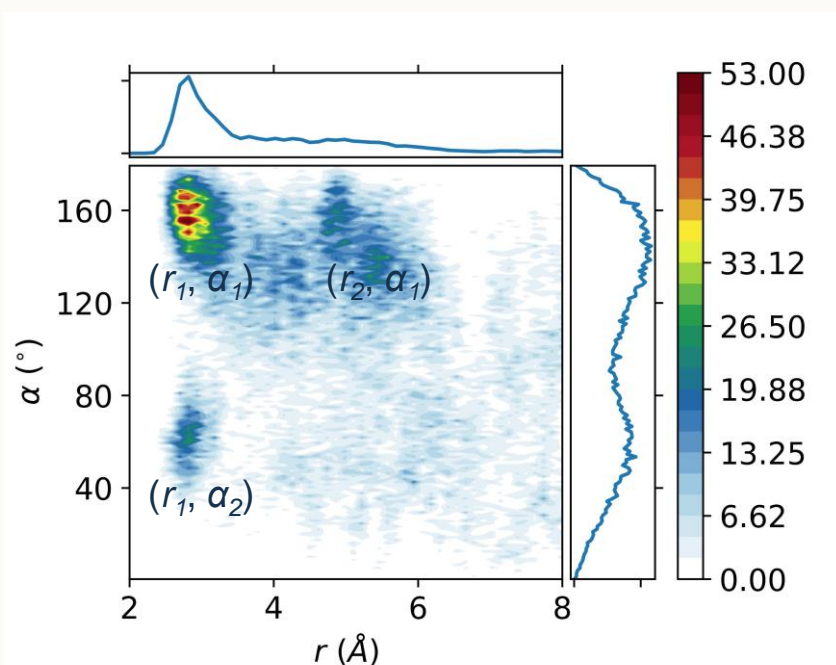
Example: water diffusion in supramolecular cages

Acceptor:
TFA Os

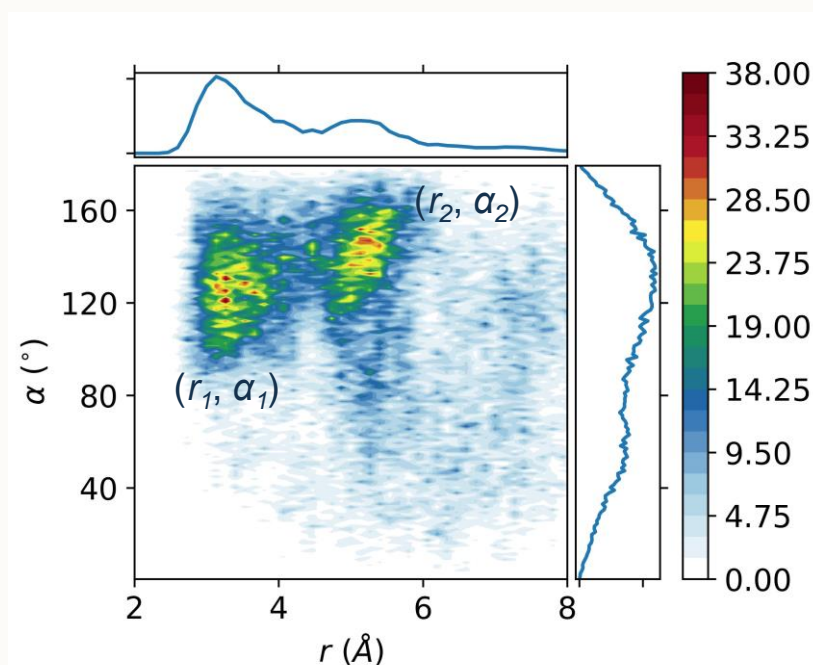
Sub-centres due to
equivalent H/O and
rotational dynamics



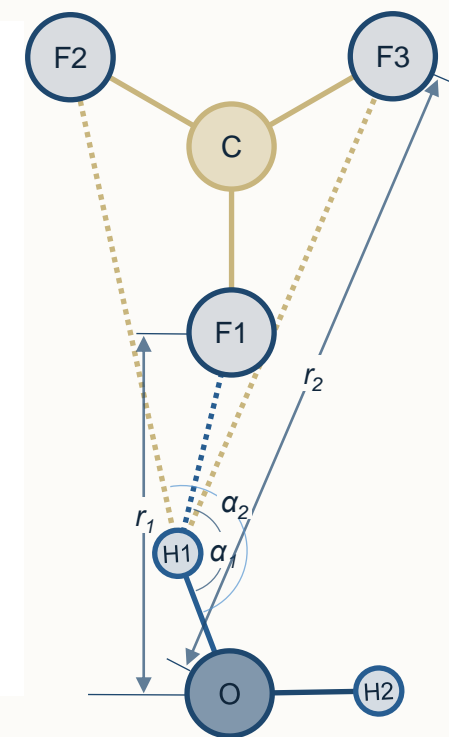
Donor: water OH



Less distinct centres
due to weaker H bonds
and rotational dynamics



Acceptor:
TFA Fs



Donor: water OH



Example: water diffusion in supramolecular cages

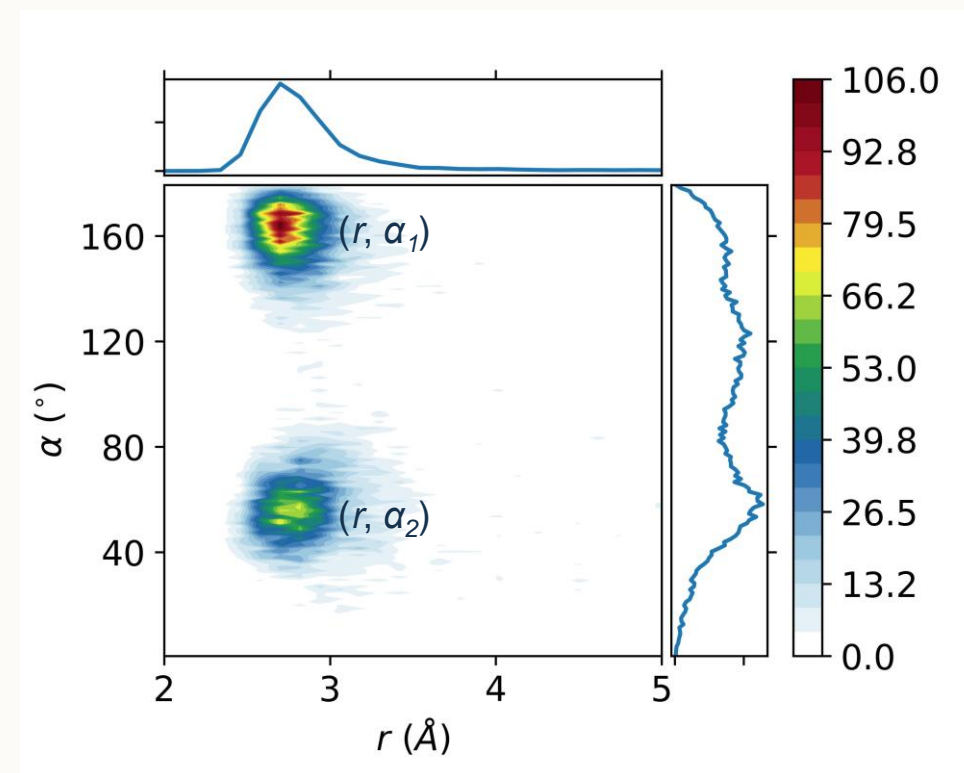
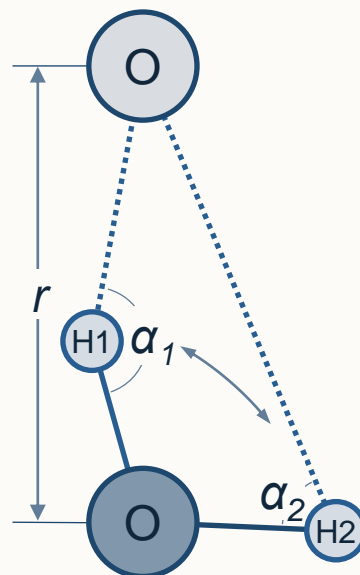
Water clustering

Donor: water OH

Acceptor: water O

Two distinct centres due to equivalent positions

Potential to be as proton conductive as Nafion





Preparation for practical session

- Copy files from shared folder:
`cp /work/ta154/shared/cp2k_md_training.tar.gz .`
- Unzip the files:
`tar -xvf cp2k_md_training.tar.gz`
- Open a new tunnelling session (check your port_num from the port.txt file):
`grep name port.txt`
`ssh -L ****:localhost:**** username@login.archer2.ac.uk`
- [Load python module for data analysis:](#)
`cd /work/ta154/ta154/username`
`module load PrgEnv-gnu`
`module load cray-python`
`python -m venv --system-site-packages /work/ta154/ta154/username/mdana`
`source /work/ta154/ta154/username/mdana/bin/activate`
`pip install nglview ase`
`cd cp2k_md_training && jupyter notebook --no-browser --port=****`



Practical session I

AIMD-equilibration

- Input file and script preparation
- Submit equilibration job

- Files:
 - mg.xyz
 - sys.inp
 - job.slurm
- Submit
 - sbatch job.slurm



Practical session I

AIMD-production job

- Check energy and temperature
- Prepare files, change settings
- Submit job
- Data analysis using provided trajectory data

Restart from equilibration

- Create new folder:
 - `mkdir ensemble-temp`
- Files:
 - `cp sys-1.restart ensemble-temp`
 - modify the settings:
ENSEMBLE, THERMOSTAT, TIMECON and add
EXT_RESTART session
 - `cp job.slurm ensemble-temp`
- Submit
`sbatch job.slurm`



Force evaluation in CP2K: machine learning based methods

Supported machine learning based molecular dynamics:

- [Nequip and Allegro](#)
- [Neural Network Potentials](#)
- [PAO-ML](#)
- [DeePMD-kit](#)



Force evaluation in CP2K: NNP

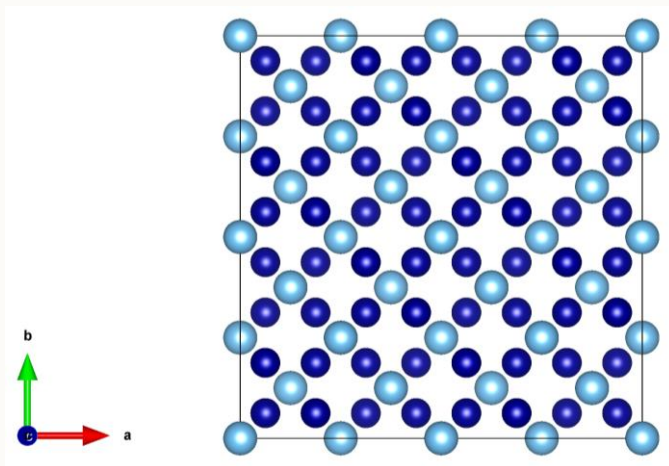
NNP: Parinello-Behler Neural Network Potential (PBNNP)

Atomic
positions

Features

Neural
Network

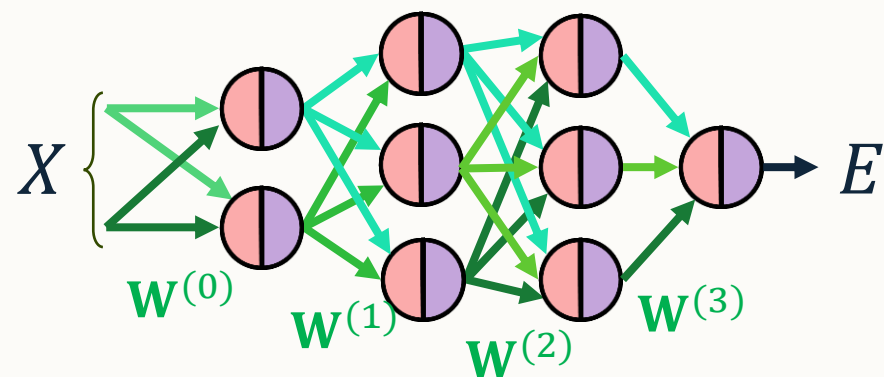
Energy
Prediction



Parinello-Behler
descriptor

$$G_{RAD} = \sum_{i \neq j} F[r_{ij}]$$

$$G_{ANG} = \sum_{\substack{i, k \neq j \\ j < k}} F[r_{ij}, r_{jk}, r_{ik}]$$

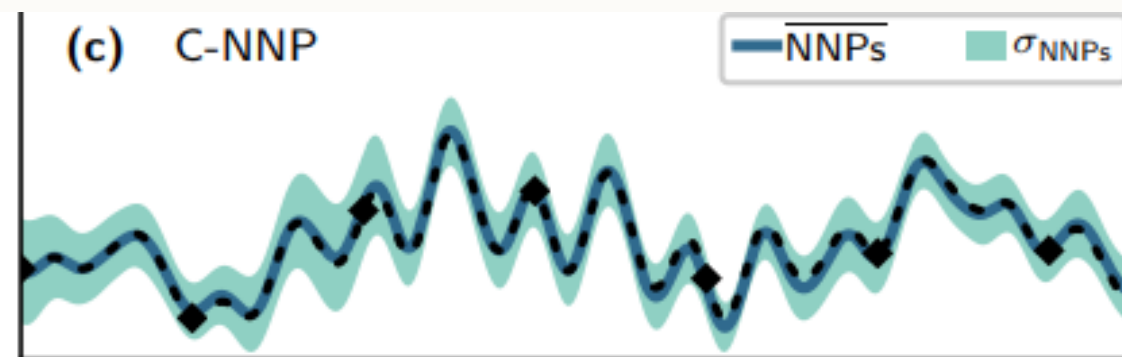
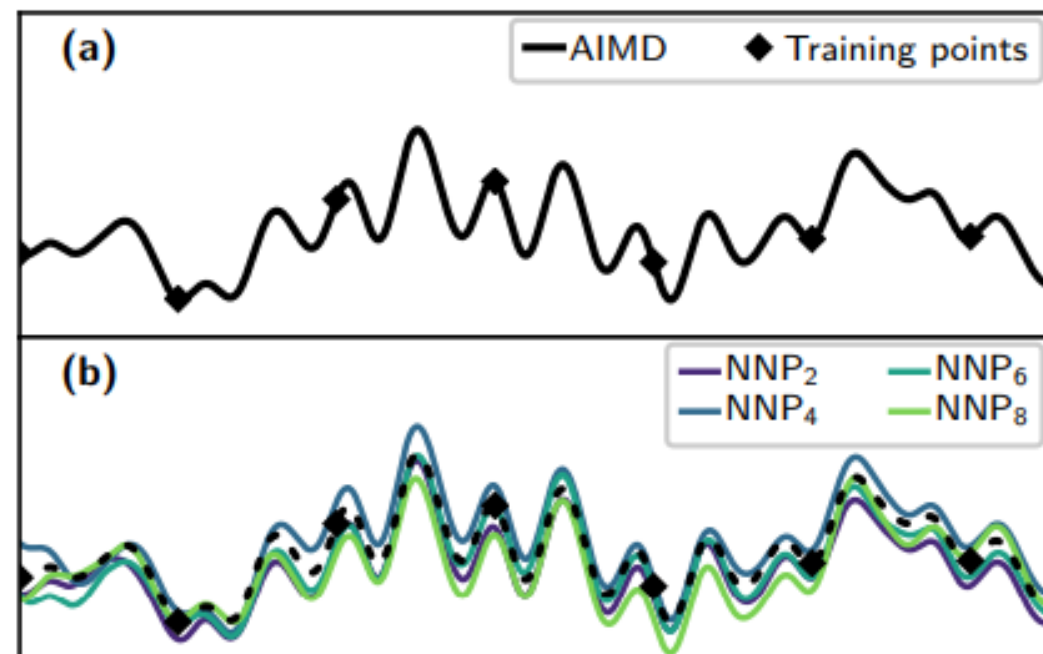
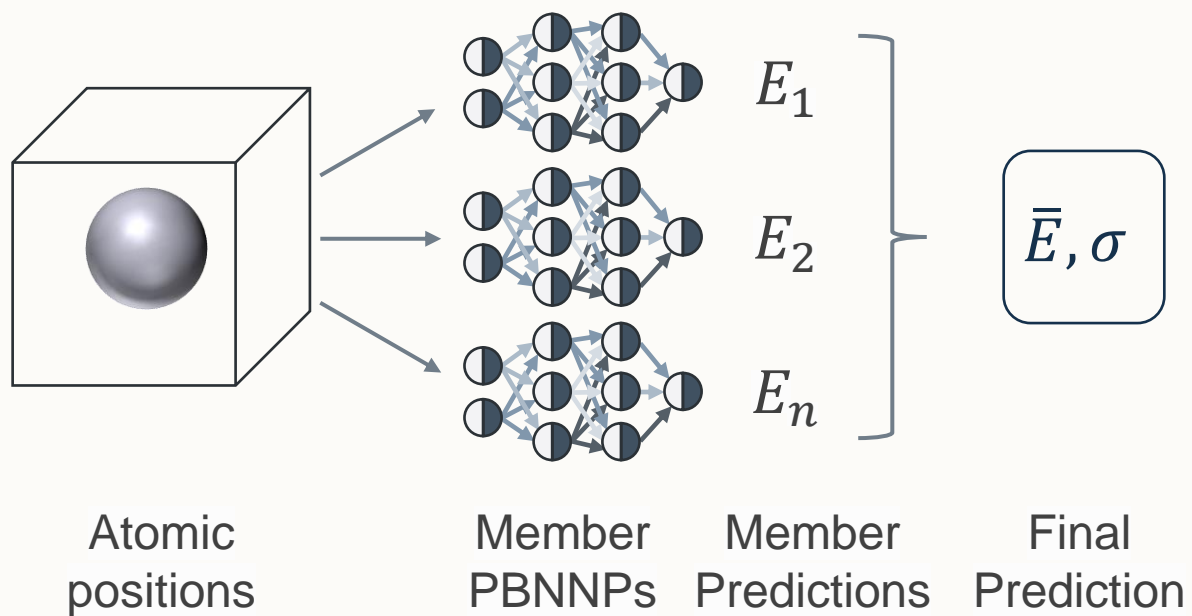




Why NNP?

Committee Neural Network Potentials:

- Reduce data usage
- Improve accuracy





Training data from CP2K AIMD data

Use [AML](#) to convert AIMD data generated by CP2K to N2P2 format

Note: the original package takes constant cell parameters only, which means NPT data cannot be correctly converted

A copy of modified AML which read and write changing cell data is provided, refer to the `Prepare_N2P2_training_data.ipynb` Notebook on usage



Force evaluation using NNP

NNP: neural network potentials

```
&FORCE_EVAL
```

```
METHOD NNP
```

Use NNP to calculate force

```
STRESS_TENSOR ANALYTICAL
```

Include this if using NPT

```
&NNP
```

```
NNP_INPUT_FILE_NAME ${PATH}/input.nn
```

```
SCALE_FILE_NAME ${PATH}/scaling.data
```

```
&MODEL
```

NNP file paths here

```
WEIGHTS ${PATH}/weights
```

```
&END MODEL
```

```
&END FORCE_EVAL
```




Practical session II

MLMD

- MLIP MD: equilibration
- MLIP MD production run
- Compare time taken / step
- Compare the $g(r)$ obtained by AIMD and MLMD

- Use pretrained committee NNPs
- Files
 - input script
 - job submission script
 - NNP files
- Submit
- Use the Jupyter Notebook to plot your results.

Further readings

- CP2K
 - [CP2K manual](#)
 - [CP2K useful tools](#)
- Thermostats:
 - [Brief introduction to the thermostats](#)
 - [MD Ensembles and Thermostats](#)
- Spectroscopy
 - [Infrared spectra](#)
 - [Vibrational spectroscopy](#)

- Machine learned interatomic potentials
 - [Machine Learning Force Fields](#)
 - [Applications and Advances in Machine Learning Force Fields](#)

Also, check out CP2K exercises (where some contents on MD and thermostats are taken from):

[exercises \[CP2K Open Source Molecular Dynamics\]](#)