RASPA Workshop/School Exercises Day 2

3 September 2024

Note: Always examine all the input- and output files in detail. Options are explained in the provided manual.

Exercise 1: Radial Distribution Function (RDF) in a fluid

Spatial distributions like the RDF contain a considerable amount of information about the local structure and the correlations between molecules. This pair correlation describes the coordination structure of the fluid, which is indicated by pronounced peaks in the RDF. Many observables are directly related to the RDF, in particular the neutron or X-ray scattering intensities as a function of wave vector (i.e. structure factors). Thermodynamic quantities such as energy, pressure or chemical potential can also be described in terms of the RDF. Moreover, by integrating the RDF under the first peak, we can obtain the coordinate number (number of particles coordinating a given particle in its first solvation shell). To define the RDF, $g(r)$, the configurational distribution function is integrated over the positions of all atoms except two. The function gives the probability of finding a pair of atoms at a distance r apart, relative to the probability expected for a completely random distribution at the same density.

$$g^{(n)}(\mathbf{r}_1,\ldots,\mathbf{r}_n) = \frac{V^n N!}{N^n\,(N-n)!}\frac{1}{Z_{\mathrm{NVT}}}\int\cdots\int \exp\left[-\beta U(\mathbf{r}_{n+1},\ldots,\mathbf{r}_N)\right] d\mathbf{r}_3,\ldots,\mathbf{r}_N \qquad (1)$$

where N is the number of particles, V the volume, ρ the average number density N/V, β the inverse temperature, Z_{NVT} the canonical partition function, $\mathbf{r}_i$ the positions of the particles (with $i = 1,\ldots,N$), $U(\mathbf{r}_1,\ldots,\mathbf{r}_N)$ the potential energy between the particles, $\mathbf{r}_{ij}$ the separation vector between particle i and j. The normalisation factor ensures that $g(r\to\infty) = 1$, but periodic boundary conditions limit the large r values to less than half the smallest perpendicular distance of the simulation box to avoid the interaction of particles with multiple periodic images of their neighbouring particles. For isotropic and homogeneous fluids we have for the n-order correlation function Of special interest is the second-order correlation function

$$g(r) = \frac{V}{N(N-1)}\left\langle \sum_i \sum_{j\neq i} \delta\left(\mathbf{r}-\mathbf{r}_{ij}\right)\right\rangle \text{ where } \quad \mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j \qquad (2)$$

For an ideal gas we have

$$g(r) = \frac{N-1}{N} \qquad (3)$$

The RDF computation is turned on by adding a few lines to a standard Molecular Dynamics input:

```
"SimulationType" : "MolecularDynamics",
"NumberOfCycles" : 100000000,
"NumberOfInitializationCycles" : 1000,
"NumberOfEquilibrationCycles" : 10000,
"PrintEvery" : 10000,

"Systems" : [

    "Type" : "Box",
    "BoxLengths" : [30.0, 30.0, 30.0],
    "ChargeMethod" : "None",
    "CutOff" : 12.0,
    "ExternalTemperature" : 300.0,
    "Ensemble" : "NVT",

    "ComputeConventionalRDF" : true,
    "WriteConventionalRDFEvery" : 5000,
    "SampleConventionalRDFEvery" : 10,
    "NumberOfBinsConventionalRDF": 128,
    "RangeConventionalRDF" : 15.0,

    "ComputeMSD" : true,
    "SampleMSDEvery" : 10,
    "WriteMSDEvery" : 5000

],

"Components" : [

    "Name" : "methane",
    "TranslationProbability" : 0.5,
    "RotationProbability" : 0.5,
    "ReinsertionProbability" : 0.5,
    "CreateNumberOfMolecules" : 100

]
```

Question:

1. Set the number of cycles low, and run the code to make a movie to view it. What do you observe?

2. Run the code and compute the RDF at 100, 200 and 400 molecules per simulation box. What do the features in the RDF mean? Why is the low density and the high density curve very different?

3. Can you guess how the RDF gives any guidance in how to choose an appropriate cutoff? What happens if one chooses a cutoff smaller a distance where $g(r)$ is not yet unity?

4. Could you also have used Monte Carlo to compute the RDF?

5. How much slower is the molecular simulation of the 200 molecules per simulation box compared to the 100 molecules per simulation box?

Exercise 2: Self-diffusion in a fluid

In MD simulations, successive configurations of the system are generated by integrating Newton's laws of motion, which then yields a trajectory that describes the positions, velocities and accelerations of the particles as they vary with time. The self-diffusivity describes the motion

of individual particles. In an equilibrium molecular dynamics simulation the self-diffusion coefficient D_α of component α is computed by taking the slope of the mean-squared displacement at long times

$$D_\alpha = \frac{1}{2dN_\alpha} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \sum_{i=1}^{N_\alpha} \left(\mathbf{r}_i^\alpha(t) - \mathbf{r}_i^\alpha(0)\right)^2 \right\rangle \tag{4}$$

where N_α is the number of molecules of component α, d is the spatial dimension of the system, t is the time, and $\mathbf{r}_i^\alpha$ is the center-of-mass of molecule i of component α.

The Einstein and Green-Kubo equations given above can be applied to each x, y, z-direction individually (when the dimension of the system is taken in each case as $d = 1$), applied to the two dimensional case $d = 2$, or applied to the three dimensional system $d = 3$. In this case the directionally averaged diffusion coefficient is given by

$$D = \frac{D_x + D_y + D_z}{3} \tag{5}$$

The conventional algorithm to measure autocorrelation functions can be implemented in several ways. Figure 1 shows the general framework for computing any kind of correlation function. In this example, time indices 80 to 95 are shown. First, we need a buffer to store the correlation function. The size of this buffer is chosen in advance and thus limits the correlation function to a predefined maximum time interval. At time index 80, an origin is stored. The data after the origin are then correlated with the origin. In this simplified example the buffer is of size 10, so the maximum correlation time is 9 Δt, where $\Delta t = \delta t \times \tau$ is the integration time step δt times the sampling interval τ. The sampling interval τ is taken as 1 in Fig. 1 but it is often 5-10 integration steps in practice. After 10 time sampling steps, the buffer is full. In general, the time average of a property A in a simulation is computed as $\langle A \rangle = \frac{1}{N} \sum_{i=1}^{N} A_i$. Here, A_i is the current buffer that is full. We keep track of the summation of all these values during the simulation in an array denoted as the *accumulation buffer* (not shown in the figure) and a counter N keeps track of the number of buffers added. The average correlation function can be plotted at the end of the run or anytime during the run by printing the accumulated buffer divided by the counter. After the update of the accumulated buffer, a new time origin is stored, and the process is repeated. An important improvement is the use of overlapping buffers. In the example, not just one, but three buffers are used, each with a different offset in time. This offset is evenly spaced. Each time step contributes multiple data points to the various buffers and therefore improves the efficiency of the algorithm. Ideally, the overlap should be confined to time intervals over which the correlation between measurements has vanished, i.e. using 10 buffers in this example does not produce the *maximum* improvement in accuracy because successive samples of the buffers are usually correlated.

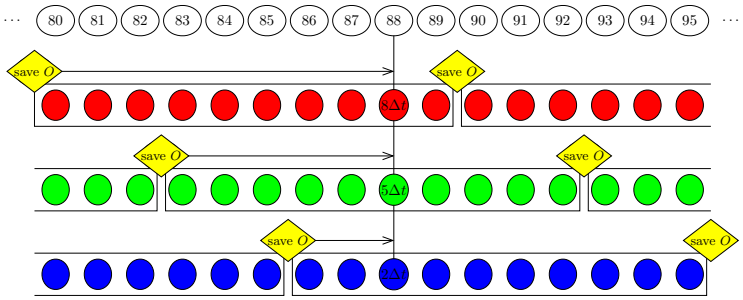


Figure 1: Conventional technique to sample correlation functions. Shown here are simulation steps 80 through 95 and 3 buffers of size 10 with different origins in time that are in simultaneous use. The time origins of the blocks are evenly spaced. The current simulation step is 88. At this step, the current value is combined with the stored origin of buffer 1 to compute the correlation at a time difference of $8\Delta t$. But it is also combined with the two other blocks for time differences of $5\Delta t$ and $2\Delta t$, respectively. The use of multiple buffers increases efficiency. Each of the buffers contains, when full, a sample of the ACF. At the end of the next time step 89, buffer 1 is full. The ACF is added to an array containing the accumulated ACFs (not shown in the figure), and the value at step 90 is stored as the new origin O. In real applications, about 10-20 buffers of size 100-500 are usually used, and often the sampling is only every 5-10 MD steps.

Each of the buffers contains a full sample of the autocorrelation function, which is added to the accumulated ACF when completely filled. An alternative is to store just the origins (instead of the buffers), and add the contribution to the accumulated ACF immediately while keeping track with an additional array of the amount of times a contribution has been added *per index*. This adds one array, but the need for storing the buffers is removed.

Conventional methods to measure correlation functions are unable to measure fast and slow decay simultaneously. This limitation arises due to the fixed sampling frequency. A high sample frequency leads to large memory requirements as well as high cpu-time demands. With a low sampling frequency one can obtain long-time correlations, but any fast decay will be missed. The *order-n algorithm* by Frenkel and Smit allows for an adjustable sampling frequency, and fast and slow decay can be sampled simultaneously at minimal computational cost.

Recipe to get diffusion-coefficients from MSD-plots:

- Plot MSD in log-scale and find the time-value where the MSD has really become linear.
- fit `f(x)=6*D*x+b` for isotropic diffusion (`using 1:2`) or `f(x)=2*D*x+b` for direction diffusion (`using 1:3` for x, `using 1:4` for y, and `using 1:5` for z).
- Use a small fitting range. Values at longer times are less and less accurate.

For example, in gnuplot:

```
plot "msd/msd_self_methane.s0.data" us 1:3 title 'x-direction'
f(x)=2*D*x+b
fit [1000:5000] f(x) "msd/msd_self_methane.s0.data" us 1:3 via D, b
plot "msd/msd_self_methane.s0.data" us 1:3 title 'x-direction',f(x)
```

Using the `fit` command, and a range (here from 1000 to 5000 picoseconds), the diffusion coefficients can be extracted from the linear fit. The obtained number is in terms of 10^{-8} m^2/s.

1. For this exercise nothing needs to be run, the output of the previous exercise can be used (in addition to the RDF, also the MSD was actually calculated). Using for example `gnuplot`, plot the MSD in `msd/msd_self_methane.s0.data`. What do the columns in the file mean? Try to obtain diffusion coefficients from the slopes of the MSD.

2. Why does the MSD start with a slope of 2 and eventually after long times reaches a slope of unity?

3. What do you observe for collective (center of mass) diffusion? Why?

Exercise 3: Self-and collective diffusion of methane in MFI

We are going to compute self-diffusion and collective diffusion of methane in MFI-type zeolite. The loading is 32 methane molecules in total, so 4 molecule per unit cell. To compute dynamic properties, MD must be used and a thermostat (Nosé-Hoover) to control the temperature.

For a single adsorbed component, the transport diffusion coefficient D^T is given by

$$D^T = \frac{\Gamma}{2dN} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \left(\sum_{i=1}^{N} \left(r_i\left(t\right) - r_i\left(0\right) \right) \right)^2 \right\rangle \tag{6}$$

The thermodynamic factor Γ is

$$\Gamma = \left(\frac{\partial \ln f}{\partial \ln c} \right)_T \tag{7}$$

and can be obtained from the adsorption isotherm. Isotherms can be obtained experimentally or predicted from GCMC simulations. Alternatively, Γ can be computed during a GCMC simulation as

$$\Gamma = \frac{\langle N \rangle}{\langle N^2 \rangle - \langle N \rangle^2} \tag{8}$$

The omission of the thermodynamic factor in Eqs. 6 leads to the corrected diffusivity D^C

$$D^C = \frac{1}{2dN} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \left(\sum_{i=1}^{N} \left(r_i\left(t\right) - r_i\left(0\right) \right) \right)^2 \right\rangle \tag{9}$$

The corrected diffusivity is in physics also known as *collective diffusion*. This type of diffusion is a property that depends on the total system and can be viewed as a center-of-mass diffusion.

1. Think about what you would expect diffusion to behave like in this system. Do you think diffusion is equal in all directions?

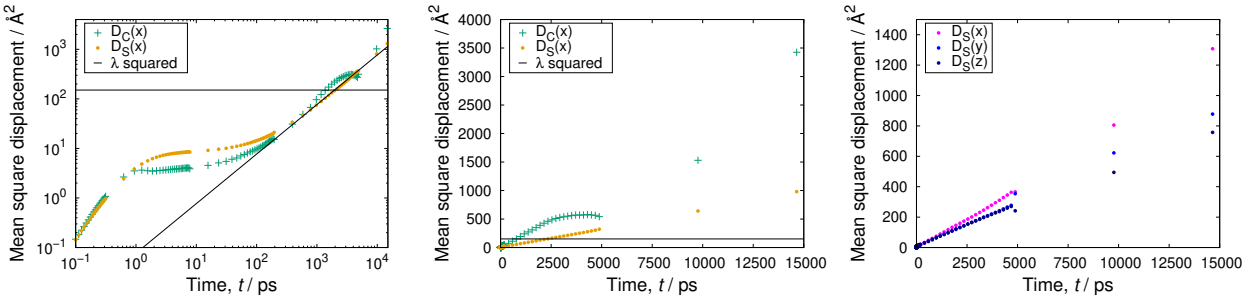
2. How much slower do you think diffusion in confinement is compare to the fluid?

3. Do you think self-diffusion is faster, slower, or the same as collective diffusion? Why?

4. Start the MSD simulation. Keep it running and go to directory `msd` and plot using `gnuplot` the files `msd_methane.s0.data.dat`. Plot the data in both normal scale and in log-log-scale. Explain the behavior of the MSD.

5. Why is collective diffusion so much harder to compute than self-diffusion?

Exercise 4: Self-and collective diffusion of methane in ITQ-29



(a) Log-log scale (b) Normal scale (c) Normal scale

Figure 2: Mean-squared displacement (MSD) of methane in ITQ-29 at 300K, (a) Log-log scale, (b) Normal scale, (c) Normal scale, self-diffusion in x, y, z.

1. Diffusion of methane in ITQ-29 is slow. Too slow to get results in a few hours. Therefore the results have already been run for you. Examine the MSD's. Based on this, what would you estimate as the limit (in m^2/s) that MD can be used for?

2. Take a diffusion coefficient of 10^{-15} m^2/s and require that a molecule travels to the next cage (distance 10 Å). The time-step of the simulation is 1 fs. How many MD steps does one need?

3. How can you compute slow diffusion?

References

[1] D. Frenkel and B. Smit. *Understanding Molecular Simulation 2nd edition.* Academic Press, London, UK, 2002.

[2] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids.* Clarendon Press, Oxford, 2017.

[3] B. Dünweg and K. Kremer. Molecular dynamics simulation of a polymer chain in solution. *J. Chem. Phys.*, 99(9):6983–6997, 1993.

[4] I.-C. Yeh and G. Hummer. System-size dependence of diffusion coefficients and viscosities from molecular dynamics simulations with periodic boundary conditions. *J. Phys. Chem. B*, 108(40):15873–15879, 2004.

[5] O.A. Moultos, Y. Zhang, I.N. Tsimpanogiannis, I.G.Economou, and E.J. Maginn. System-size corrections for self-diffusion coefficients calculated from molecular dynamics simulations: The case of co_2, n-alkanes, and poly(ethylene glycol) dimethyl ethers. *J. Chem. Phys.*, 145(7):074109, 2016.

[6] S.H. Jamali, L. Wolff, T.M. Becker, A. Bardow, T.J.H. Vlugt, and O.A. Moultos. Finite-size effects of binary mutual diffusion coefficients from molecular dynamics. *J. Chem. Theory Comput.*, 14(5):2667–2677, 2018.

[7] Y. Zhang, A. Otani, and E.J. Maginn. Reliable viscosity calculation from equilibrium molecular dynamics simulations: A time decomposition method. *J. Chem. Theory Comput.*, 11(8):3537–3546, 2018.