at the University of Freiburg

Master Thesis

March 30, 2021

Author:

Wilkin Wöhler

Supervision:

Tanja Schilling

Abstract

A raw writing of hard sphere nucleation, a simulation to measure quantiteis, and a analysis of data generate by means of the simulation. A test citation: [**Heyes2007**]

Contents

List of Figures

List of Tables

1 Introduction

1.1 Hard sphere system

Some test text.

The Hard Sphere system is the simplest model of a fluid including interactions between the single particles. Its well known potential between particles i and j reads:

$$V(r_{ij}) = \infty \cdot \Theta(\sigma - r_{ij}) \tag{1.1.1}$$

In this equation r_{ij} indicates the distance between the two particles, σ is the diameter of the Hard Spheres and Θ is the Heavyside function.
While the ideal gas model without pair interactions already makes it possible to derive famous equations as $pV = NkT$, it does not include phase transitions yet. But these can be observed when granting the particles to take up space. Because it is the simplest model and it is well feasible for computer simulations the Hard Sphere system is very well suited to study basic properties of phase transitions.

Compared to experiments where similar systems can also be realized, general properties of the system at hand can be varied very precisely without much effort and position data of the single particles can be extracted easily as well, because they naturally are required for the simulation.

On the downside computer simulations are much more constraint in their size, but with todays computational possibilites system of the order of 1 million particles become tractable, and such computer simulations become a powerfull tool to study also phase transitions in simple systems.

The beginning of such simulations actually dates back to the beginning of electronic computer technology (cite Alder and Wainwright 1959). Since then more algorithms to increase efficency have been elaborated, and technology advanced giving today the possibility of studying large systems.

1.2 metastable fluid/ phase diagram

The equation of state for the simple Hard Sphere system has various approximations, (cite an overview paper), The probably most common approximation due to its simplicity is the Caranhan-Sterling approximation:

$$Z = \frac{1 + \eta + \eta^2 - \eta^3}{(1-\eta)^3} \tag{1.2.1}$$

(cite https://aip.scitation.org/doi/10.1063/1.1672048) It approximates the compressibility factor Z depending on the packing fraction η for the Hard Sphere fluid.

For the stable brach after nucleation a common approximation is given by the Alamrza equation of state(cite https://aip.scitation.org/doi/full/10.1063/1.3133328).

$$\frac{p(v-v_0)}{k_B T} = 3 - 1.807846y + 11.56350y^2 + 141.6y^3 - 2609.26y^4 + 19328.09y^5 \tag{1.2.2}$$

where p is the pressure, v is the volume per particle $v_0 = \sigma^3/\sqrt{2}$ is the volume per particle at close packing, including the diameter of the spheres σ, and $y = p\sigma^3/(k_B T)$, where k_B is the Boltzman constant and T is the temperature of the crystal.

From these two equations of state we can draw the phase diagram: (include grapic of phase diagram.)

The chemical potential difference between the two equations of state can be calculate from the difference between the two equations of state.

Eventhough not further discussed in this thesis it might be said that for polydisperse radii the phase diagram becomes even richer as show for example in https://journals.aps.org/prl/pdf/10.1103/PhysRevLett.91.06830

1.3 Classical nucleation theory :/

Classical nucleation theory (CNT) has been proposed look who did this the first time and since then multiple times modified to describe various types of systems. Eventhough its predictions often deviate far from experimental results. look if tanja cited someone else here. Either way it can provide a reference to know what to expect roughly and also it can be checked by the simulation data.

CNT assumes that a spherical crystalite may form in the liquid with properties of the bulk crystal while the fluid remains with the properties of the bulk liquid. The difference in the free energy landscape is given by a surface and a volume term. The first arises from the surface tension γ between the fluid bulk and solid bulk phases. The second comes by the difference in chemical potential $\Delta\mu$. The whole expression reads:

$$\beta\Delta G(R) = 4\pi R\gamma - \frac{4}{3}\pi R^3\rho\Delta\mu \tag{1.3.1}$$

Where ρ is the particle density of the solid phase.

The free energy barrier can be sketched like this (maybe sketch it for som mu and gamma).

As can be seen the free energy landscape of eq. 1.3.1 has a maximum at a radius called R_{crit}. The interpretaion of this radius is that if a cluster, which is defined as a structure having a crystalline like ordering locally, surpasses the critical radius, it is likely to continue to grow until it incorporates all available fluid. This critical radius is given by eq. 1.3.2.

$$R_{crit} = \frac{2\gamma}{\rho\Delta\mu} \tag{1.3.2}$$

Furthermore the height of this barrier can be calculated to be

$$\beta\Delta G(R_{crit}) = \frac{16\pi\gamma^3}{3\rho^2(\Delta\mu)^2} \tag{1.3.3}$$

In the classical picture now we look at an activated process which has an rate given by the Arhenius law:

$$k = c\exp(-\frac{k_BT}{\Delta E}) \tag{1.3.4}$$

$$\Leftrightarrow \quad k = c\exp -\beta\Delta G(R_{crit})\text{THIS HAS TO BE CORECCTED} \tag{1.3.5}$$

As the constant c is not further defined the absolute nucleation rate in this picture is not predicted but only set into realtion to ther rates. Check carefully in how far this is correct!
Given the equation of state for the liquid and fluid phase from section 1.2, and taking a literature value for γ cite the refernce to which the value corresponds we can calculate R_{crit} for various densities. Plot of r_{crit} for different densities

1.4 Memory including approaches

CNT assumes a Markovian system, which means that it diffuses through a free energy landscape without granting it memory. But it has been shown by Kuhnbold ... cite Anjas Lennard-Jones System that for a Lennard-Jones system memory effects can not be neglected if an acurate desciption is desired. Such it must be assumed that also for the Hard Sphere system memory effects may be present.

To analyze such memory effects a framework by Hugues Meyer(cite Hugues) has been elaborated. In

it a memory kernel for coarse-grained observables by means of projection operators is defined together with an efficent algorithm to calculate such.
With the derived memory kernel an equation of motion for the observable can be dervied, ressmbling mostly the Generalized Langevin equationbu is called non-stationary Generalized Langevin Equation because the memory kernel can depend not only on $K(t-\tau)$ but instead on two times $K(t,\tau)$:

$$\frac{dA_t}{dt} = \omega(t)A_t + \int_0^t d\tau K(\tau,t)A_\tau + \eta_t \quad , \tag{1.4.1}$$

For the Markovian process the memory kernel is approximately given by a Dirac delta functional $\delta(\tau - t)$, in which case the Langevin equation is recovered.

1.5 Computer Precision

The precision of computer certainly influences the outcome of simulations. (talk with fabian if he found any papers reagarding varying results fromvarying precision over time)
And it should always be kept in mind that the simulation only approximates the real world. Even smallest variations in the last digits of positions, changes the simulation radical after a certain number of steps. This comes by the fact that we face a complex system with chaotic behaviour, which means that even small variations will grow exponentally until the system has nothing in common anymore. Look if you can visulaize such an behaviour by As it can be seen after x timesteps the two system have radicallly changed.

1.6 Comparsion to Real world experiments

Since the 1950s ? people have systhesized hard sphere like systems in the lab. Today a whole zoo of systems is known. All of these system have in common, that the hard spheres are in a bath of a fluid, which surrounds them. This fluids density and optical refractive index shhould match the density and optical refractive index of the hard spheres to prevent segmentation and to enable optical measurements of the system. maybe elaborate a little more on why each of the two is necessary

The absence of the bath in simple hard sphere simulations is probably the largest difference to the hard sphere systems in the laboratory. It has been argued that the difference can be circumvented by normalizing with a diffusion length or time, but a discussion on the possibility of hydrondynamic effects changing the behaviour of the lab system compared to simulations is ongoing at the moment.

For simulations it is much harder to include the bath becuase it introudces a multiple number of particles, making it very slow to simulate large systems.

2 Simulation details

During the coarse of the master thesis an event driven molecular dynamic (EDMD) simulation code has been elaborated. The choice to use the EDMD approach is taken because interst in the actual dynamics of the system were desired. This means that simulations probing the phase space of the system instead of the dynamics, like Monte Carlo (MC) simulation schemes, are not suited.
Furthermore the discontinous potential of the hard spheres is an obstacle not easy to face in regular molecular dynamics (MD) schemes, where the Newtonian equation of motion for the particles is numerically integrated.
The EDMD approach on the other site actually requires these discontinuities as will be discussed in the following sections, together with some details of the program.

2.1 Algorithm and Simulation details

In this section we will highlight the main differences to regular MD simulations, as they are the main tool to otherwise probe the dynamics of the system. Furthermore we will stick to the hard sphere example when discussing the EDMD simulations, but it can be kept in mind that the EDMD approach also allows to simulate particles with other potentials as long as the potentials are only containing step functions.

The decisive difference between EDMD simulations and regular MD schemesis that, instead of evaluating all pair and external forces on each particle and then evolving the whole system to the next time step, EDMD simulations do not have a predefined time step, but the system is evolved from one event to the next one. An event in this context is defined as the time where the next collision in the whole system takes place.

The event prediction algorithm is follows closely the approach proposed by Bannerman et. al [**Bannerman2014**] which will be discussed in the next section.

2.1.1 Event driven molecular dynamics (EDMD)

For the prediction of events in EDMD simulations an overlap function $f_{ij}(t)$ between particles i and j is defined, where the squared quantities are used merely because they are easily accessible.

$$f_{ij}(t) := |\vec{r}_j(t) - \vec{r}_i(t)|^2 - \sigma^2 \tag{2.1.1}$$

$$\left| \begin{array}{ll} \text{with} & \vec{r}_i(t) = \vec{r}_i(t_0) + (t - t_0)\,\vec{v}_i(t_0)\,, \\ & \Delta t := t - t_0\,, \\ & \vec{v}_{ij}(t) := \vec{v}_j(t) - \vec{v}_i(t)\,, \\ & \vec{r}_{ij}(t) := \vec{r}_j(t) - \vec{r}_i(t)\,, \\ \Leftrightarrow & \vec{r}_{ij}(t) = \vec{r}_{ij}(t_0) + \Delta t\,\vec{v}_{ij}(t_0) \end{array} \right. \tag{2.1.2}$$

$$f(t) = (\vec{r}_{ij}(t_0) + \Delta t\,\vec{v}_{ij}(t_0))^2 - \sigma^2 \tag{2.1.3}$$

$$f(t) = |\vec{r}_{ij}(t_0)|^2 + \Delta t^2\,|\vec{v}_{ij}(t_0)|^2 - 2\Delta t\,\vec{r}_{ij}(t_0)\cdot\vec{v}_{ij}(t_0) - \sigma^2 \tag{2.1.4}$$

The overlap function has the property that it is negative for two particles being closer than their diameter, 0 for at collision and positive if neither overlapping nor touching. The calculation of the next collison thus is to calculate the roots of eq. 2.1.4.

Solving for Δt with $|\vec{r}_{ij}(t_0)|^2 := rr$, $|\vec{v}_{ij}(t_0)|^2 := vv$ and $\vec{r}_{ij}(t_0)\cdot\vec{v}_{ij}(t_0) := rv$ is rather trivial:

$$0 = rr + vv\,\Delta t^2 - 2rv\,\Delta t - \sigma^2 \tag{2.1.5}$$

$$\Leftrightarrow \quad 0 = \Delta t^2 - \frac{2rv}{vv}\,\Delta t + \frac{rr - \sigma^2}{vv} \tag{2.1.6}$$

$$\Leftrightarrow \; \Delta t = -\frac{rv}{vv} \pm \sqrt{\left(\frac{rv}{vv}\right)^2 - \frac{rr - \sigma^2}{vv}} \tag{2.1.7}$$

But a caveat when executing on a floating point machine is present as can be seen when considering which solution is of interest. As for a possible collision it is necessary that the two particles move towards each other we can conclude that the scalar product is required to be negative $rv < 0$, because otherwise the particles are already moving away from each other.

Also the quadratic formula has two solutions, corresponding to the entry and the exit of the overlap. Because the entry has to be prior to the exit, we further conclude that interest lies on the smaller solution that is:

$$\Delta t = \frac{-rv - \sqrt{(rv)^2 - vv(rr - \sigma^2)}}{vv} \tag{2.1.8}$$

Now for the case where the distance of the spheres is already close to the diameter of the spheres we find $(rv)^2 \gg (rr - \sigma^2)$, which results in a cancelation of two large numbers leaving a small number. Floating point number operations are inherently bad suited because they tend to large inaccuracy in this case. Rewriting eq. 2.1.8 by making use of the third binomial formula look if this is fine to write. leads to:

$$\Delta t = \frac{(rr - \sigma^2)}{-rv + \sqrt{(rv)^2 - vv(rr - \sigma^2)}} \tag{2.1.9}$$

Comparably eq. 2.6.1 does not contain a cancelation of the type seen before and such is better suited for the use in a computer simulation. cite Goldberg '91

The event prediction algorithm proposed by Bannermann[**Bannerman2014**] works by differentiating 4 cases:

1. If $rv > 0$ the particles move away from each other leading to a collision time of $\Delta t = \infty$.
2. If $rr < \sigma^2$ an overlap is present resulting in an immediate collision time of $\Delta t = 0$
3. If $(rv)^2 - vv(rr - \sigma^2) \leq 0$ the two particles miss each other, including touching without momentum transfer, resulting in a collision time of $\Delta t = \infty$
4. If none of the before is given the particles collide and Δt is calculated by eq. 2.6.1.

All collision times for a particle are then stored in a queue sorted by event time called particle event list (PEL). From the PEL the first entry is then passed to the global FEL.
This procedure initally takes place for all particles to set up the system and later on for particles involved in an event after its execution.

As will be discussed in section 2.1.2 some widely used measures like reducing redundant calculations or implementing a cell system to reach $\mathcal{O}(N)$ computation timehave been implemented.

A further detail to take care of is the possibility of scheduled events which have bcome invalid due to a earlier collision of the one of the particles. This is handeled by assigning an interaction count to each particle and then store this at precalculation time with the event. When the event comes up, and the interaction count of one of the particles has increased in the meantime, the event is said to be invalidated. Depending on which particle had an event in the meantime the invalidation either causes no action or a recalculation of new events.

2.1.2 Details of the Implementation

As the simulation code is based on an earlier Monte Carlo Code for hard spheres a complete walk through the whole progam would become quite extensive. Such we will focus on key points to understand the details of the simulation.

Event **struct**

We start with the basic *Event* struct which includes 6 entries as shown in tab. 2.1.1. The type of *time*

Datatype	**Name of entry**
(timeType)	time
(int)	event_type
(particle*)	particle
(void*)	partner
(int)	particle_count
(int)	partner_count

Table 2.1.1: Content of the *Event* struct.

(timeType) is usually set to double. The *time* variable itself represents the time for when the event is scheduled.
The *event_type* variable is either set to 0 or 1 and indicates if the event is a cell transfer or a collision of two particles.
The *particle* variable is a pointer to the particle for which the event has been pre calculated, while the *partner* variable is set to be a void pointer. Such it is possible to either interpret it as a particle pointer for the collision type event or as an interger pointer to the index in the current cells' neighbours list for transfer events.
In the last two rows the interaction counts for particle and partner are listed as well. As the destination cell in a transfer event does not require an interaction count, the *partner_count* variable is only used for collision events.

The *event* struct is used for all events throughout the simulation. For read and write operations with the HDF5 file format, the struct *event_data* is available which uses only indexes instead of pointers.

Particle **class**

The *Particle* class is comparably to the one from the MC code basis. Its MC related variables have been removed and additional key variables and concepts will be discussed in the following:

First a vector storing events called *backupEvent* has been added to make it possible to store events from the pre calculation for the case of the first event being invalidated. The idea of reusing events is discussed in many publications, for example that the memory cost increases onyl moderately with more backup events while the speedup does not increase much for more than two stored events [**Bannerman2011**]. It also has been argued that the added complexity can not account for the increase in efficency[**DONEV2005**].
Eventhough in the own simulations a decrease in calculation time of more than 10% was observed and the cost of complexity was seen as moderate. The difference might be explained by the fact that the systems under consideration in this thesis have a rather large particle density, leading to more invalid collsions.

In the context of reusing pre calculated results, it should also be mentioned that after a cell transfer the recalculation of events can be restricted to partner particles only in the new neighbouring cells, leading to only 1/3 of the calculation time in this case. But as mentioned systems under consideration are mostly rather dense and such the number of transfers is often at below 5%. Thus the increase in efficency was assumed to be to costly on the complexity side, and not implemented. Eventhough for sparse systems, it might make sense to include an *updatePEl* routine.

Also key differences to the former MC Particle type are the variables *total_interactions* and *particle_delayed_time*. The first is the variable for book keeping of interactions, while the second represents the event driven character of the simulation. Because each particle only moves on purely ballistic trajectories until an event occurs, it is not necessary to keep all particle positions and velocities synchronized in time. Quite on the contrary it would mean executing extra operations together with summing rounding errors by each floating point operation.

Because sometimes it is desired to have the whole configuration at one point of time, the *transferToTime()* function of the particle provides the possibility to take the particle into the present. This is necessary soon as measurements are performed on the system, including snapshots.

As mentioned before the system behaves chaotic even under slightest changes like a rounding error from an extra floating point operation. A result of this is that measuring at different rates during a simulation changes the simulation trajectory quite a bit. It has been observed that such a system may keep close to the undisturbed trajectory for about 50-100 events/particle. As it is of desire to measure quantities and take snapshots without disturbing the simulation, the simulation program makes employs copies of the configuration being costly in terms of memory but making simulation resets or higher sampling rates at interesting points possible within a defined trajectory.

The structure is as follows: The first copy is rewritten with an image of the working configuration just before any measurement. The working trajectory iteself is then disturbed by the measurement, and afterwards replaced with its state before the measurement from the backup configuration.
The second copy actually includes the full simulation state, while the first only includes the particle cofiguration. This second one might be used to save a state during the simulation and reset to just the same point at any later time.

The *Box* class

The box of the simulation stayed mostly the same as in the previous MC code. One chanege is the array *neighbours_lookup* which has been added. It contains the indices for the cells' *neighbours* array pointing to cells that share their surface. It is used to identify which cell a particle has to be transfered to during a cell transfer event.

Furthermore the *Update* routine now takes care of all quantities depening on the length of the box, making the *rescale* routine a simple rescaling of the edge lengths with an additional *Update* command.

The *Scheduler* class

While the afore mentioned parts of the program are necessary for the EDMD program, the *Scheduler* class contains the most distinct parts of the program. It keeps track of all events to come, predicts new events and orchestrates the execution of the events. The essential functions are discussed in the following subsections while some basic properties are shortly highlighted here.

First of all the *Scheduler* holds the Future event list (FEL) in which at least one event per particle is stored. As discussed within section 2.1.2 the simulation is capable of saving the complete state of a trajectory, including all pre calculated events. For this purpose an array of *Events* is available.
Furthermore the *Scheduler* includes the *gloabl_time*.
Important for the efficency is the pre allocation of all arrays used within the prediction calculations, as the number of executions for the collision prediction routine is about $\frac{30}{\text{particle}\cdot\text{step}}$ accounting to a few billion function calls during a small simulation.

Scheduler::predictTransfer()

As the name suggests this function predicts the next cell transfer of a particle due to its movement. For this it calculates the position of the particle at global time, which for a valid state of the simulation always lies within its cell. By transforming the momentary position of the particle from the global coordinate system to the coordinate system of the cell and taking into account the periodic boundary conditions, we can write for each dimension i the equations

$$t_{i1} = -\frac{r_i}{v_i} \quad \text{and} \quad t_{i2} = \frac{l_i - r_i}{v_i} \tag{2.1.10}$$

which describe the times when the particle pierces the cell's left and right boundaries. A negative time corresponds in this case to a boundary crossing in the past, a time comparable to 0 means that the particle is on the edge of its cell and a positive time means that the boundary crossing lies in the future. By going through the different possible cases for t_1 and t_2 we find the resulting next crossing time for each case as shown in tab. 2.1.2.

t_1	t_2	Result	Case
$>$	$>$	invalid	-
$>$	$=$	$t_{\text{crossing}} = t_1$	0
$>$	$<$	$t_{\text{crossing}} = t_1$	1
$=$	$>$	$t_{\text{crossing}} = t_2$	2
$=$	$=$	invalid	-
$=$	$<$	$t_{\text{crossing}} = t_1$	3
$<$	$>$	$t_{\text{crossing}} = t_2$	4
$<$	$=$	$t_{\text{crossing}} = t_2$	5
$<$	$<$	invalid	-

Table 2.1.2: Possible results for left and right crossing time with resulting choice of next crossing time. $>$, $=$ and $<$ are to be read as for example $t_1 > 0$. The case indicates the case number within the actual simulation.

By collecting the next crossing times for each dimension and taking the minimum of these times the exit time of the particle from its cell is determined.

The return value of the routine is an *Event* where the partner is given as an address to the box' *neighbours_lookup*. The index lies between 0 and 5, corresponding to the 6 possible neigbhour cells sharing a surface with the current cell of the particle. Each valid case represents a distinct neigbour cell and the index within the cells *neighbours* array is clearly defined by the cell setup routines and is shown in tab. 2.1.3.

dimension	boundary	case	index
x	front	0	12
	back	1	13
y	front	2	10
	back	3	15
z	front	4	4
	back	5	21

Table 2.1.3: Overview of the cells' *neighbours* indices directly sharing a surface for 3 dimensions. As the indices hardly follow any simple pattern they are explicitly noted at this point. Obviously the cell consists of a front and a back boundary in each dimension. The corresponding case matches the one from tab. 2.1.2.

Scheduler::predictCollision()

The prediction of collision times has already been discussed in section 2.1.1. The implementation in the program first calculates all necessary scalar products while accounting for the periodic boundary conditions, and in a second step returns the collision time depending on the case at hand.

The presented algorithm is only valid for single sized particles. If polydisperse systems are supposed to be considered the algorithm has to be adjusted. mayhap do it in the appendix?

As this routine is executed through out the simulation very often it has been tried to optimize its efficency multiple times. For example calculating only necessary results for the next case differentation has been implemented but no significant increase in efficency was recognized and for better readability the prior version has been reestablished. In either case if an optimized way of calculating the results is found it might be useful to use them.<– Not really nice

Scheduler::setupFEL()

This routine fill the FEL of the simulation. For this purpose it iterates through all particles and calls *setupPEL* for each of them. The PEL in turn is set up by predicting the next cell transfer as well as the next collisions with all particles within the 3^d cells directly surrounding the particle. From all predicted events only such with finite times are then written to the *backupEvents* vector corresponding to the PEL of the particle.
For the FEL only the top event of each particle is then used. Because other events from the PEL are able to move on to the FEL ounce written to the FEL an event has to be erased from the PEL.

Scheduler::executeTransfer()

The execution of a transfer event is accomplishd by the event particles *MoveBetweenCells()* routine. The departure cell is taken as the event particles own cell. While the event partner holds the information which of the cell neighbours is the destination cell.

Scheduler::executeCollision()

The outcome of a collision between particle 1 and 2 with corresponding position and velocity can be derived by momentum and energy conervation cite some textbook, or better jsut write down the calculation too be

$$\vec{v}_1' = \vec{v}_1 + \left(\frac{\vec{r} \cdot \vec{v}}{\vec{r} \cdot \vec{r}} \right) \vec{r}_{12} \tag{2.1.11}$$

While eq. 2.1.11 is not directly depending on the radius, an indirect dependence by the collision time and configuration at which eq. 2.1.11 is evaluated persists.

Scheduler::executeEvent()

The execution of an event works in multiple steps. At first the topmost *Event* is copied from the FEL where it is deleted. Next the validity of the interaction counts of both particle (*cond1*) and its partner (*cond2*) are evaluated. The validation is nothing else than a comparison of the interaction counts when the event was scheduled with the present interaction counts. As the conditions are used in the following flow statements they are stored as boolean type. Furthermore in the case of a transfer event the validation of the partner is not necessary but only for better readability performed.
It follows a distinction between 5 cases which are given by:

Valid transfer (*event_type*==0 and *cond1*)
: The transfer is executed, the global time is evolved to the event time and the particles PEL is rebuilt and its next event pushed to the FEL.

Valid collision (*event_type*==1 and *cond1* and *cond2*)
: The collision is executed, the global time is evolved to the event time and for both particpating particles new PEL's are built and each top event is pushed to the FEL.

Invalid transfer (*event_type*==0 and not *cond1*)
: The particle must have had an interaction previously where a new event for it was scheduled, thus no action is taken.

Invalid collision due to particle (*event_type*==1 and not *cond1*)
: The particle must have had an interaction previously where a new event for it was scheduled,

thus no action is taken as for the above.

Invalid collision due to partner (*event_type*==1 and not *cond2*)
Only the partner had an interaction previously where a new event for it was scheduled, thus a new event for the particle is required. As the particle had no further interactions, the *backupEvents* are still valid and its first entry is pushed into the FEL. In case no backupEvents are stored the PEL is rebuilt and its first entry pushed to the FEL instead.

The order of the cases might be exchanged, except for the last two. This is because the last one assumes *cond1* to be true, which is guaranteed by the case before.

Furthermore the routine counts the number of each case, to monitor numbers of collisions, transfers and invalidated cases by type. This is not required by the simulation but can be helpful for understanding the system and simulation.

2.2 Measurement quantities

list quantities and their use etc.

2.2.1 Mean squared displacement

-Averaged or as distribution
-Reuqired for diffusion

2.2.2 Radial distribution fucntion

-Well show an RDF?

2.2.3 Largest Cluster

- Non local
- large in case of nucleation
- requires almost no memory

2.2.4 Cluster size distribution

- Non local
- requires more memory still not as much as for example particle positions

-

2.2.5 Cluster positions and numbers

- not done
- local as single clusters can be named and followed from formation to melting or final growth
- Much more information as locallity can be archieved.

2.2.6 Tensor of Gyration

-Only done for largest cluster of size larger than some threshold
-How to extract the eigenvalues l1,l2,l3
-Radius of Gyration
-Anisotropy
-Asphericity

2.3 Probe of simulation code

To probe the simulation dynamics we measuremed the longtime diffusion constant and the radial distribution function of the stable hard sphere liquid, as there are many measurements available in the literature to compare with.

2.3.1 Diffusive behaviour

The diffusive behaviour of particles in a liquid usually can be three sepearted in thee distinct parts. First the short time diffusion which can be understood as the random movement of the particles within their momentary cage within the fluid. Second a sub diffusive phase in which the particles are repelled for the first time by their nearest neighbours. And third the long time diffusion to describe the random propagation of the particle through the fluid over time.

As the ballistic hard sphere system enters into the long time diffusion almost at ounce only this is really measureable. By assuming a diffusive process we have the expectaion that the average mean squared displacement (MSD) of a particle can be well described by

$$\langle x^2 \rangle(t) = 2\, d\, D\, t\,, \tag{2.3.1}$$

where $\langle x^2 \rangle$ is the expectation value of the MSD, d the number of spatial dimensions, D the characteristic diffusion constant, and t the time by which the system has evolved.

By measuring $\langle x^2 \rangle (t)$ and making a linear regression to the data points we can find the Diffusion constant D.

To probe the simulation code, systems as characterized in tab. 2.3.1 have been used. The equilibration phase has been carried out up to $\eta_f = 50\%$ at the final volume fraction while above $\eta_f = 50\%$ an inital volume fraction of $\eta_i = 45\%$ has been used to obtain a fluid rather than a solid after the equilibration phase. As the measurement stretches into the metastable regime, it has also been necessary to check for nucleation events which were not present during the measurements.

The resulting diffusion constants depending on the fluid volume fraction are shown in fig. 2.3.1 alongside values obtained from the literature, for a similar system.

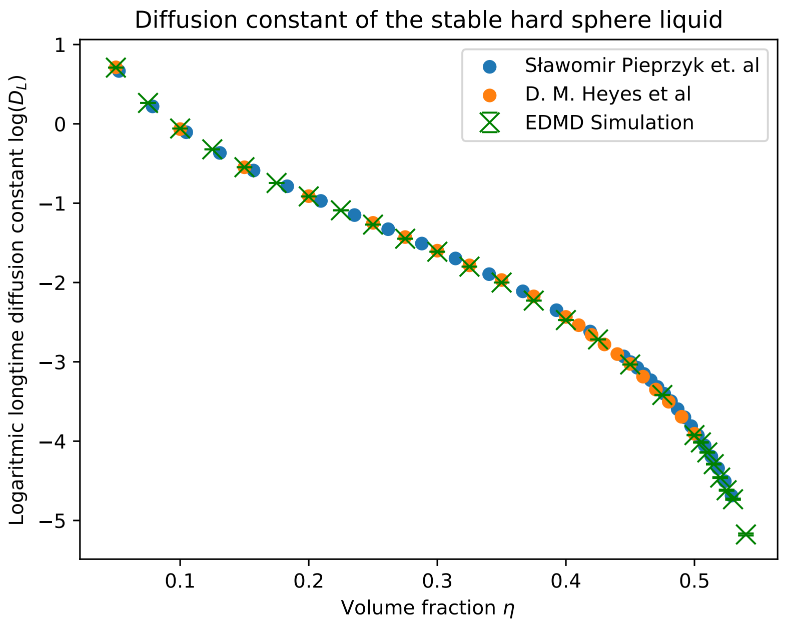


Figure 2.3.1: Logarithmic long time diffusion constant of the hard sphere liquid as measured in the own simulations as well as measurements from the literature. cite the people.

As it can be seen the EDMD simulation is very well capable of reproducing the diffusion constant for the hard sphere liquid, and such we expect the dynamics of it to accurately represent the purely ballistic hard sphere system.

Parameter	Value
N	16384
eq_steps/particle	5000
pr_steps/particle	20000
η_i	5% ... 50 %
η_f	5% ... 54 %

Table 2.3.1: Input parameters of diffusion test systems.

2.3.2 Radial distribution function

A further well known quantity for the hard sphere system is the radial distribution function. As a theoretical prediction the Percus-Yevick approximation can be used to compare with, also it would be possible to compare with Monte Carlo simulations of the hard sphere system. In fig. 2.3.2 an overview for a range of volume fractions is shown from the same simulations used in section 2.3.1. Clearly visible is that no particles enter within the diameter of the spheres. Further for higher volume fractions the liquid shells become very well visible. At very high volume fraction we also find that new peak arises below $r = 2\sigma$.

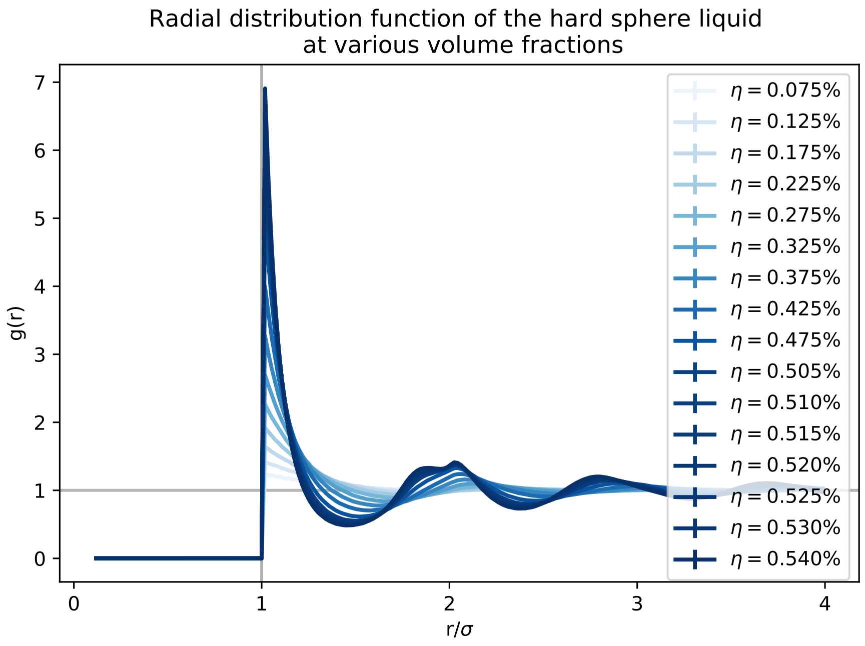


Figure 2.3.2: Radial distribution functions for a range of volume fractions. The colouring corresponds to the used volume fraction.

To compare with Percus-Yevick approximation the radial distirbution function for two single volume fractions is shown with the corresponding theoretical solution in fig. 2.3.3.

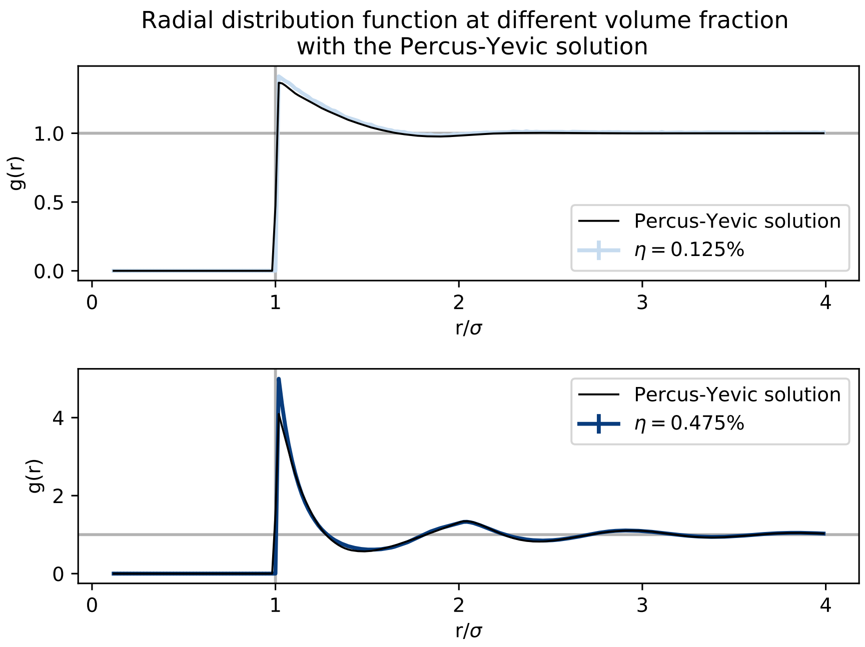


Figure 2.3.3: Radial distribution function for the hard sphere system at a low and at a high volume fraction of the liquid together with the theoretical prediction from the Percus-Yevick approximation.

As highlighted for example in [**Hansen2006**] the theoretical approximation has some flaws as can be seen with $g(r)|_{r=1\sigma}$ being too low for the Percus-Yevick approximation. Eventhough we see that overall the two radial distribution functions follow each other rather closely.
Such we are confident that the elaborated simulation code is capable of producing accurate data in other contexts as well.

It seems in the hansen and mcdonal that the peak below 2 sigma is also not present in their MC g(r), is there a reason for this?

2.4 Estimate of required resources

To choose system parameters senseful calculation times and file sizes of the simulation were characterized. This was of interest as the program was supposed to run on the NEMO high performance computing cluster which puts hard boundaries on calculation times which when treepassed can cause tremendous loss of data if not corretly caught by the program.

2.4.1 Calculation time estimates

The calculation time of the program was tested for a large range of different system sizes up to almost 9 million particles in the fluid state. As can be seen in fig. 2.4.1 the calculation time increases proportional to the system size for the execution of a step as well as for a measurement of the fluid system. The calculation cost being of $\mathcal{O}(N)$ enables the study of large systems. Furthermore from the slope an expectation for the execution time of a single event can be deduced, as well as an expectation for the time necessary for a measurement. As discussed on the example of fig. 2.4.2 the dependence of the measurement routines on the largest cluster size were not seen here, as possile clusters remained rather small during these simulation times.

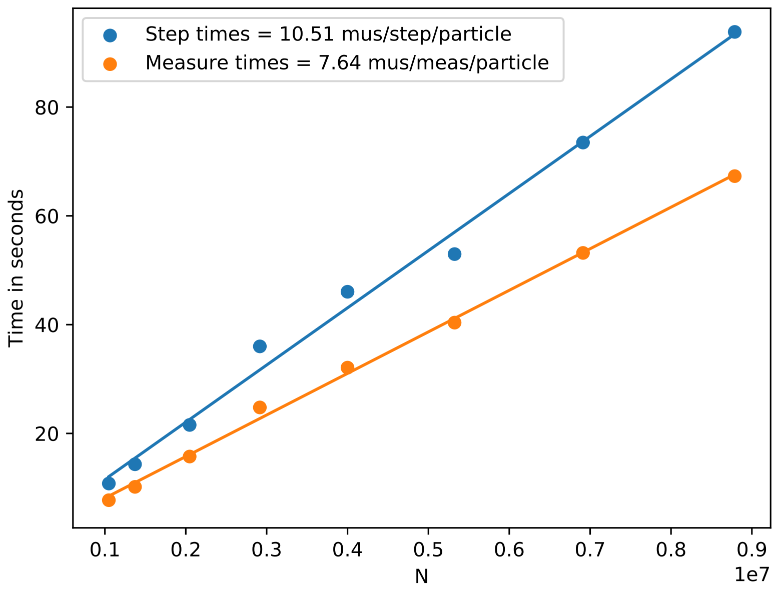


Figure 2.4.1: Overview of CPU time required for calculating a simulation step, consisting of an event for each particle, and a measurement of relevant quantities for the system. As assumed for a simulation algorithm with $\mathcal{O}(N)$ calculation effort, the data points can be described by a line rather well. As the CPU time is clearly related to the further workload of the CPU during the calculation it is also expected to find fluctuations if the other workload of the machine is not strictly controlled.

The effect of larger clusters was only investigated after problems with the runtime of the programs were traced back to these. The q6q6-order parameter routine was tested for larger clusters in a nucleating simulation with about 1 million particles within the box. As can be seen in fig. 2.4.2 the calculation cost of the cluster finding routine can be described with a quadratic dependence on the largest cluster. For an impression what this means we can use the calculation costs of a simulation step from fig. 2.4.1 being about $t_{step} \approx 10\,\mu$s/particle. Such the execution of one step takes about 10 s for 1 million particles. If a measurement is performed every 10th step, the calculation cost of the measurements without a large cluster remain below 10%. But as the largest cluster grows to a few hundred thousand particles in size, the measurements can make up 30 % and more of the calculation cost, or for a fixed number of steps, increase the calculation time by about 50 %. This previously unseen effect lead to actual data loss as the combination of NEMO cluster policy and EDMD simulation program did not result in a save shutdown of the program after breaching the walltime limit of the NEMO cluster.

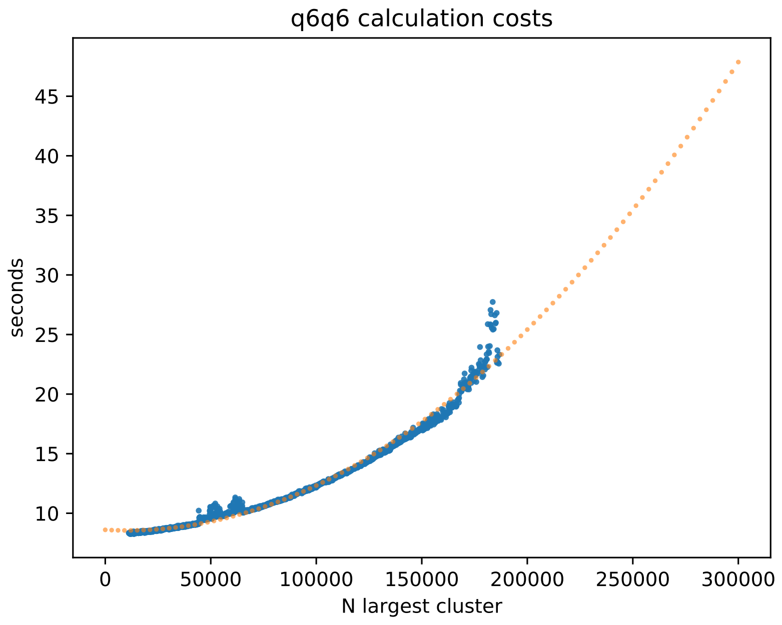


Figure 2.4.2: Calculation time of the q6q6 order parameter at an increasing largest cluster size during one nucleation, together with the quadratic best fit indicating that the q6q6 routine calculation effort can be approximated by $\mathcal{O}({N_{lc}}^2)$ where N_{lc} is the size of the largest cluster.

2.4.2 File sizes estimates

A further important constraint for the simulations are the produced amount of data. To get an impression of the filesizes, the required memory for snapshots, reset steps and other measurements were measured prior to the actual simulations. The results for a single snapshot containing all positions and velocities of all particles as well as the size of a single simulation reset step containing all positions, velocities, the FEL, all PEL's and all delayed times is shown in fig. 2.4.3. It can be seen that the file

size is directly proportional to the system size which clearly expected as each particle adds a further set of positions, velocities etc. to the saved data.
The memory costs of other measurements have been left out of fig. 2.4.3 as these only amount to substantial filesizes if measurements at about each step for long simulations are done.

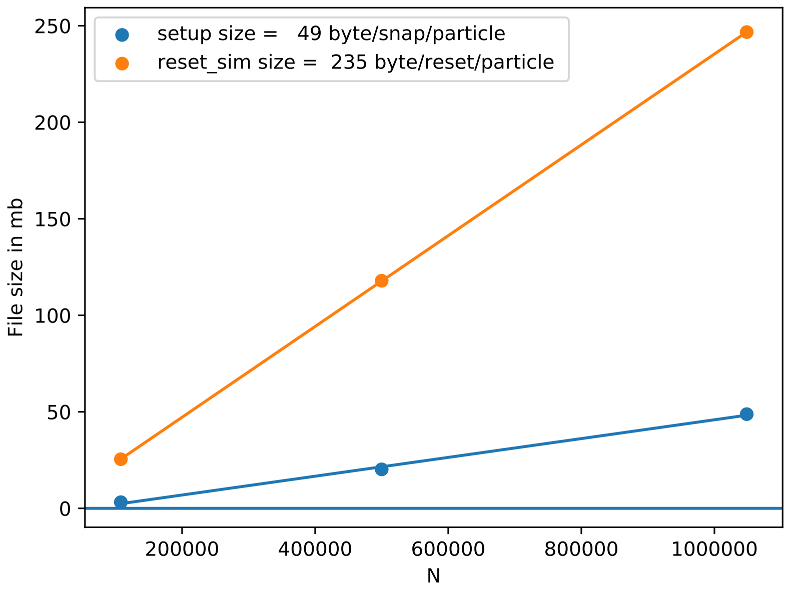


Figure 2.4.3: Overview of filesizes when a single setup on the one hand and a single full simulation on the other hand is saved for comparison reasons together with their corresponding linear regression. while the linear regression for 3 points is statistically not exceedingly senseful it still remains a useful tool to extract the slope which corresponds to the required memory per particle and snapshot or reset simulation.

2.5 First produced data

The motivation for the simulation code is based on the interest in nucleation rates of the hard sphere system at varying volume fractions. To observe a nucleation the volume fraction of hard spheres has to be changed rapidly from lower ones where the system is in the stable fluid phase to higher ones where a meta stable fluid-solid phase exists. If this metastable phase is evolved in time nucleations can be observed as stochastic distributed events. To measure those without effects originating from the handling of the simulation, some parameters were tested within reasonable ranges prior to the data production.
For this simulation the equilibration steps as well as the inital density before the volume quench

Parameter	Value
N	16384
eq_steps/particle	100 ... 20000
η_i	5% ... 49 %
η_f	54 %

Table 2.5.1: Input parameters of test systems probing the dependence on equilibration steps and inital density.

seemed like they could introduce unwanted artefacts, and thus we performed some smaller data series to evaluate if and when these effects might come into play.

The used test system is characterized by the figures in tab. 2.5.1.
The general behaviour of the systems is analysed by inspecting the cluster distribution over time. The mean cluster distribution is shown in fig. 2.5.1 together with the same data smoothed by a gaussian filter matrix. The smoothing is used because in a next step the difference between the mean cluster distribution and the cluster distributions with varying simulation parameters is compared, and without smoothing at low count rates only fluctuations are visible.

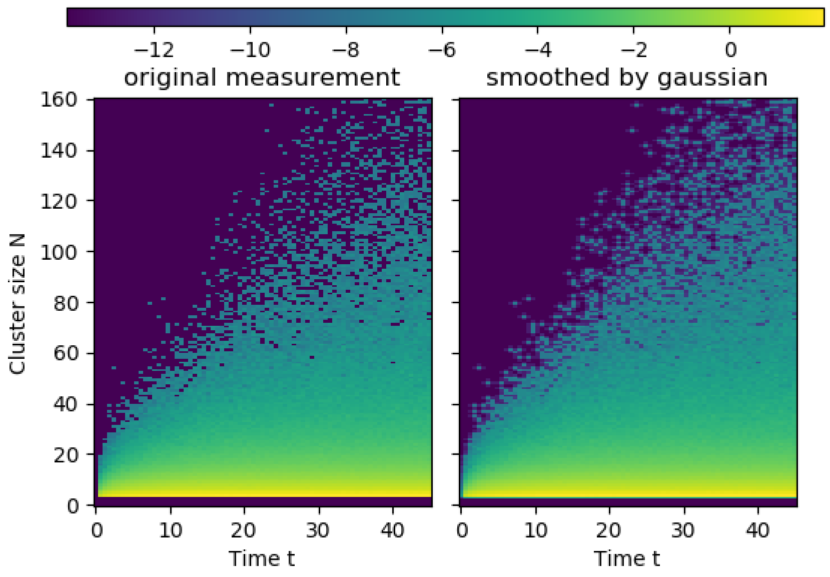


Figure 2.5.1: Heat map of the mean cluster distribution over time. The diagram encompasses 800 trajectories of 16384 particles each. The colouring indicates the logarithm of the mean cluster occurance corresponding to a probability in the stationary case.

From fig. 2.5.1 we can see how the system behaves after a volume quench into the metastable region. In the liquid rarely any clusters are present and thus directly after the quench no clusters are present either as the spatial configuration requires time to rearange into clusters. In the later evolution we see how clusters form, and soon after begin to nucleate leaving the range of the diagram.

To compare simulations with varying parameters the quantity defined in eq. 2.5.1 is used, where complications with zero values are circumvented by fixing these values below the regular signal.
Three samples of this comparison are shown in fig. 2.5.2 and fig. 2.5.2. The colouring indicates $\Delta_{p(N,t)}$ defined in eq. 2.5.1. As mentioned above the quantities $p_i(N,t)$ and $\langle p(N,t)\rangle$ have been smoothed by a gaussian filter, because the number of samples included, with 100 trajectories per series, were not sufficent to produce smooth distributions at the given sampling rate. Such without smoothing only fluctuations would be visible.

$$\Delta_{p(N,t)} = log(|\frac{p_i(N,t)}{\langle p(N,t)\rangle} - 1|) \tag{2.5.1}$$

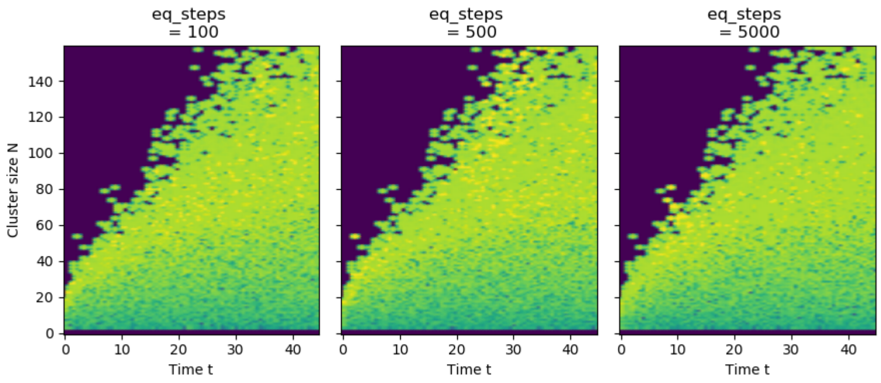


Figure 2.5.2: Heat map of differences between the cluster distributions within simulations carried out with varying the length of the equilibration phase. The quantity used for colouring is defined in eq. 2.5.1, where yellow indicates a large difference while blue indicates a small difference. Providing a legend of the colouring is ommited as $\Delta_{p(N,t)}$ has no further use as to indicate differences and actual values do not add any use.

On first sight none of them differ in their general behaviour. Because at t=0 after the quench no clusters have formed yet and also no clusters were present in the stable liquid, the difference between all simulations is zero, indicated by the blue region in the top left corner. The features visible on the edge between the zero region and the nonzero region on the other side are the same, because they are features of the mean distribution carried through. Actual differences not due to fluctuations can only be seend within the green and yellow non-zero region, but none such differences is observed.

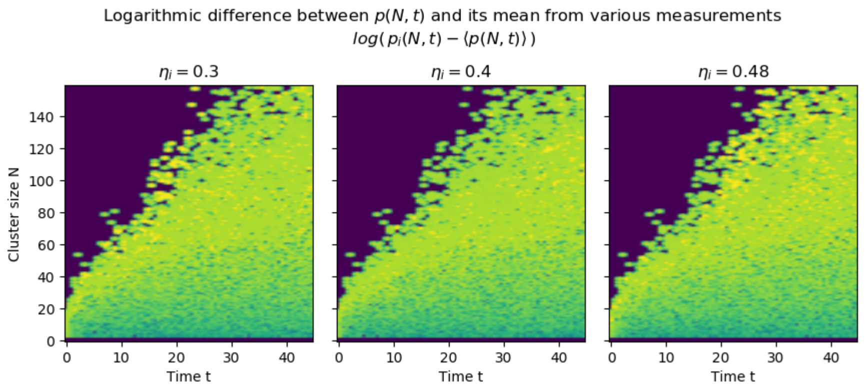


Figure 2.5.3: Heat map of differences between the cluster distributions within simulations carried out with varying the volume fraction of the liquid during the equilibration phase. The quantity used for colouring is defined in eq. 2.5.1, where yellow indicates a large difference while blue indicates a small difference. Providing a legend of the colouring is ommited as $\Delta_{p(N,t)}$ has no further use as to indicate differences and actual values do not add any use.

While it seems like the inital volume fraction of $\eta = 0.4$ and $eq\_steps = 5000$ include less irregular fluctuations, dramatic effects from choosing the simulation parameters can be excluded. Intresting in this context are esspecially the simulations with $eq\_steps = 100$ because after executing 100 events/particle on average, the inital perfect crystal configuration is only on the verge of not being detected anymore. Such one could expect that these configurations might be stoill close to crystalization, but instead we do not detect any significant difference.

A further more quantifying analysis of the differences is given by calculating the mean nucleation rates assuming classical nucleation theory. This is done for the data shown in fig. 2.5.4. The calculations of the rates have been carried out as described in section 3.7.
As we see no significant difference in the nucleation rates can be observed even for the bold setting of only 100 events per particle for the equilibration phase. Eventhough it can be seen that for this special case the rate is a little higher, but it may as well be a stochatic flucutaion.

Overall we conclude in this chapter that as long as parameters are set within reasonable boundaries, we expect not to have systematic influences of simulation parameters.

2.6 Possible extensions

The program at this state is capable of simulating large systems including compression and relaxation. Extensions can aim at either increasing complexity or size of the systems.

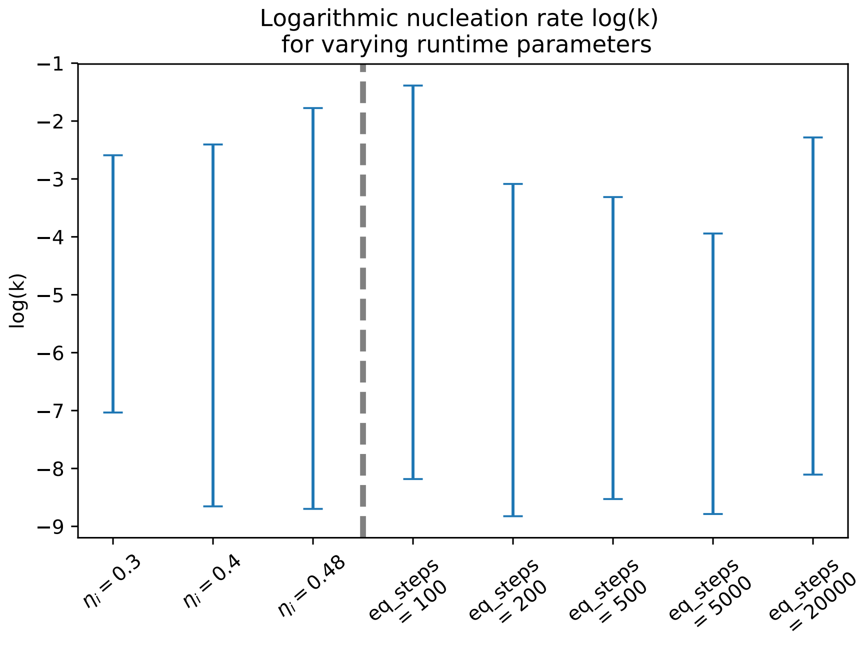


Figure 2.5.4: Comparsion of nucleation rates under CNT assumptions for different inital densities during equilibration with eq_steps fixed at 5000, as well as varying eq_steps with η_i fixed at 0.45.

2.6.1 Varying radius

A further notch of complexity would be to include polydispersity into the simulation. For this purpose the prediction of collisions has to be adjusted. When looking at the derivation of eq. 2.6.1 it is found that σ being the former diameter of a sphere in the monodisperse case has to be changed to $\sigma = R_i + R_j$. In the equations the same defiitions of scalar products are used as before in section 2.1.1.

$$\Delta t = \frac{(rr - \sigma^2)}{-rv + \sqrt{(rv)^2 - vv(rr - \sigma^2)}} \tag{2.6.1}$$

For a physical model in which the particles are made of some matter with constant density the change of the radius is also accompanied by a change of the mass. This has to be taken into account when assigning the velocities after a collision as written in eq. 2.6.2.

$$\begin{aligned} \vec{v}_i{}' &= \vec{v}_i + \frac{2m_j\,(rv)}{(m_i + m_j)\sigma^2} \cdot (\vec{r}_j - \vec{r}_i) \\ \vec{v}_j{}' &= \vec{v}_j + \frac{2m_i\,(rv)}{(m_i + m_j)\sigma^2} \cdot (\vec{r}_j - \vec{r}_i) \end{aligned} \tag{2.6.2}$$

2.6.2 Muliprocessing

It further would be possible to precalculate different events on different processors, thereby speeding up simulations. This requires the execution of events not in the strict time order used for the EDMD algorithm shown before. Instead a check should be implemented to only execute events which are spatially seperated by some distance. This might make it possible to change only little of the dynamics. Eventhough a problem conserning this idea is that the simulation code would be required to consume less memory as this would be the bottleneck of the system sizes.
Such rather large changes would be necessary to impelemnt parrallel EDMD simulations. Either way it could help making system of about 100 million particles feasible.

3 Data Analysis

3.1 Characterization of the simulated system

As an integral part of this work large scale simulations have been executed on the NEMO High performance computation cluster. For this systems as characterized in tab. 3.1.1

Parameter	Value
N	1048576
eq_steps/particle	1000
pr_steps/particle	20000 ... 60000
η_i	45.0 %
η_f	53.1% ... 53.4 %

Table 3.1.1: Input parameters of large scale simulations on the NEMO HPC cluster. The varying steps during production come by the fact, that 20000 steps were estimated to be calculated within 3 days leaving 1 day of buffer to the hard walltime limit of 4 days. Due to the increasing calculation cost of the q6q6 cluster routines for large clusters the walltime limit was still breached and without proper reset steps the datasets could not be restarted wihtout large calculation overhead as all lost data has to be replaced, and the broken reset steps within the files would have to be removed before. Such the last proper version of the files were used resulting in varying simulation lenghts but with usually a nucleation event in case of early breakdown.

The simulations consist of four series at volume fractions of $\eta = 0.531, 0.532, 0.533, 0.534$, where each series again consists of 500 trajectories. Such at each volume fraction a total amount of about half a billion particles have been simulated in the metastable fluid.

The size of the systems was choosen at the rather large size of about 1 million particles as the first aim of the simulations is to measure induction times. When using CNT as a guideline, it can be shown that the computational effort to nucleation is not significantly increased when increasing the system size. As the Calculation time per unit of simulation time is proportional to N, it is at a given volume fraction also proportional to the Volume (eq. 3.1.1).

Required calculation time per simulation time:
$$\frac{T_{CPU}}{\delta t_{Sim}} \propto N \propto V, \tag{3.1.1}$$

Expected calculation time to nucleation:
$$\langle T_{CPU} \rangle \propto \frac{T_{CPU}}{\delta t_{Sim}} \cdot \langle \tau_{Nucleation} \rangle \propto \frac{V}{V} = const. \tag{3.1.2}$$

When assuming a constant nucleation rate density, as done in CNT, we expect the nucleation time of the whole system to be proportional to the inverse of the volume eq. 3.1.1.
Following the two early proportionalities we can inspect the behaviour of the calculation time until nucleation occurs in the system. This is proportional to the product of CPU time per unit of simulation time with the expected nucleation time in units of simulation time resulting in a cancelation of the two proportionalities. Thus the size of the system is only relevant to be choosen smaller if ordering processes are important for the system. This might be the case for polydisperse systems, but in the monodisperse case the above reasoning was found to hold true.

3.2 Diffusion in the metastable liquid

Diffusion or more precisely selfdiffusion, characterizes the movement of the single particles within the system. The diffusive behaviour for many system can be subdivided into different regimes with different physical meaning.

For the ballistic hard sphere system we have an extemly short period in which most particles are freely moving without constraint, the ballistic regime. This could be resolved in the simulation by taking measurements at extreme rates, like after every event, but is not as the result could if desired also be determined by measuring the velocity distribution.

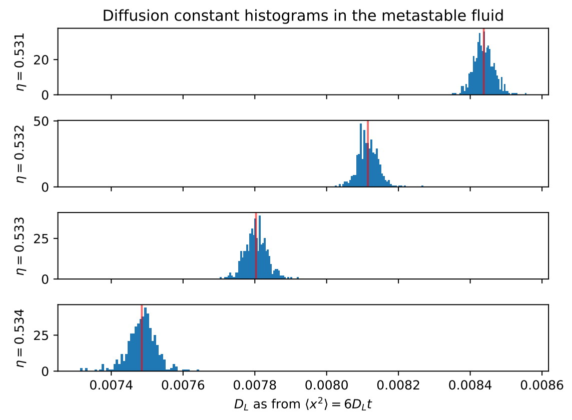
The shorttime diffusion usually seen in many systems is not seen in the ballistic hard sphere system. To understand this we can look at the physical interpretation of the short time diffusion. While it does not describe the movement of particles between different liquid cages, it only describes the browninan motion of particles in the suspension within their momentary liquid cage. As the simulated system does not contain any suspension, the shorttime diffusion is inapplicable.

The long time diffusion on the other hand is senseful to describe the movement of single particles in the simulated systems. The interpretation of the long time diffusion is that particles are able to change their momentary cage by collisions and thus can diffuse throughout the whole system until finite size effects stop their further diffusion. If circumventing finite size effects by using unwrapped coordinates the long time movement of the particles is governed by the relation eq. 3.2.1 which was first described by Einstein.

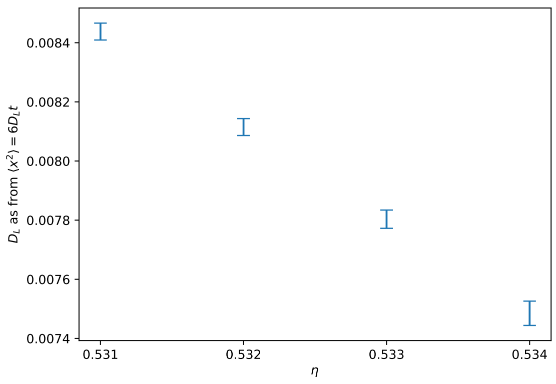
$$D_L^S = \lim_{t\to\infty} \frac{\langle(\vec{r}(t) - \vec{r}(0))^2\rangle}{2dt} \tag{3.2.1}$$

With D_L^S the longtime selfdiffusion constant which will in the following be denoted only by D, $\vec{r}(t)$ the position of a particle at time t, d the number of spatial dimensions of the system and $\langle ... \rangle$ the expectation value of the ensemble.

The average is measured in the system by saving an reference position of all particles at one point, and further caryying a set of unwrapped positions thourgh out the simulation. The average of all particles difference in their reference position with their unwrapped position is used as a measurement of the ensemble average. Esspeically for large system of 1 million particles, this quantity has only very small fluctuations as can be seen in fig. 3.2.1.



(a) Histograms of the slopes for the linear regressions to the largest clusters during the later constant growth process. The histograms are for $\eta = 0.531, 0.532, 0.533, 0.534$.



(b) Mean of the histograms with the uncertainty on the mean given by $\sigma_{\langle D \rangle} = \sigma_D/\sqrt{n}$ with n being the number of measurements included in the average.

Figure 3.2.1: Comparison of long time selfdiffusion constants at different volume fractions as histograms and their means with uncertainty.

The diffusion coefficents are important to know for comparsion between different systems. This importances is based on the idea that the fundamental mechanisms for nucleation and cluster growth do not vary between different hard sphere like systems, but are only scaled by the varying diffusion times. Furthermore their are theoretical predictions for the relationship of short time and long time diffusion, making it possible to compare experiments where the short time diffusion behaviour is better accesible with the ballistic simulations where only the long time diffusion constant is measurable.

As we see in fig. 3.2.1 the diffusion constants can be measured with rather good precision with a relative standard deviation of $\sigma_D/D \approx 1\%$. Such it does not introduce large uncertainties when normalizing time related quantities by the diffusion time $\tau_D = D^{-1}$.

3.3 Cluster growth

Ounce the clusters reach a certain size they are expected to grow with new particles being attached to the surface at a constant rate leading to a growth with a proportionality of $N \propto t^3$ as shown in eq. 3.3.1 where k is the assumed constant attachment rate, N is the number of particles in a specific cluster, A is the surface of the cluster, R is the radius of the cluster and ρ_{solid} is the bulk density which is for large clusters a good approximation of the cluster density.

$$
\begin{array}{ll}
\begin{array}{l}
\dot{N} = kA \\
\quad \left| \begin{array}{l}
\text{with} \quad N = \frac{4}{3}\pi R^3 \rho_{solid} \\
\qquad \Leftrightarrow R = \left(\frac{3N}{4\pi\rho_{solid}}\right)^{\frac{1}{3}} , \\
\text{and} \quad A = 4\pi R^2 \\
\qquad \Leftrightarrow A = \left(\frac{4\pi 3^2}{\rho_{solid}^2}\right)^{\frac{1}{3}} N^{\frac{2}{3}} ,
\end{array} \right. \\
\frac{dN}{dt} = k \left(\frac{4\pi 3^2}{\rho_{solid}^2}\right)^{\frac{1}{3}} N^{\frac{2}{3}}
\end{array}
&
\begin{array}{l}
\text{From the bottom left side} \\
\Rightarrow \quad dN\, N^{-\frac{2}{3}} = dt\, k \left(\frac{4\pi 3^2}{\rho_{solid}^2}\right)^{\frac{1}{3}} \\
\qquad \left| \ \text{setting} \quad N(t=0) = 0 \right. \\
\Leftrightarrow \quad 3N^{\frac{1}{3}} = k \left(\frac{4\pi 3^2}{\rho_{solid}^2}\right)^{\frac{1}{3}} t \\
\Leftrightarrow \quad N^{\frac{1}{3}} = k \left(\frac{4\pi}{3\rho_{solid}^2}\right)^{\frac{1}{3}} t
\end{array}
\end{array}
\tag{3.3.1}
$$

As the systems are able to accomodate clusters up to a few hundred thousand particles and usually only one very large cluster is formed during a simulation, the attachment rate can be measured by a linear regression to the third root of the number of particles in the largest cluster over time. This is visualized for the trajectories at $\eta = 0.532$ in fig. 3.3.1. The volume fraction $\eta = 0.532$ is choosen arbitrarily.

Subsequently the slopes of the linear regressions have been collected in histograms shown in fig. 3.3.2. As shown in eq. 3.3.1 these slopes correspond to constant attachment rates with a dependence on the density within the cluster. As the densities of concern are very close to each other, they only introduce a relative difference of 0.5% between the rates of lowest and highest volume fractions. Such we can neglect this dependence for the qualitative comparison.

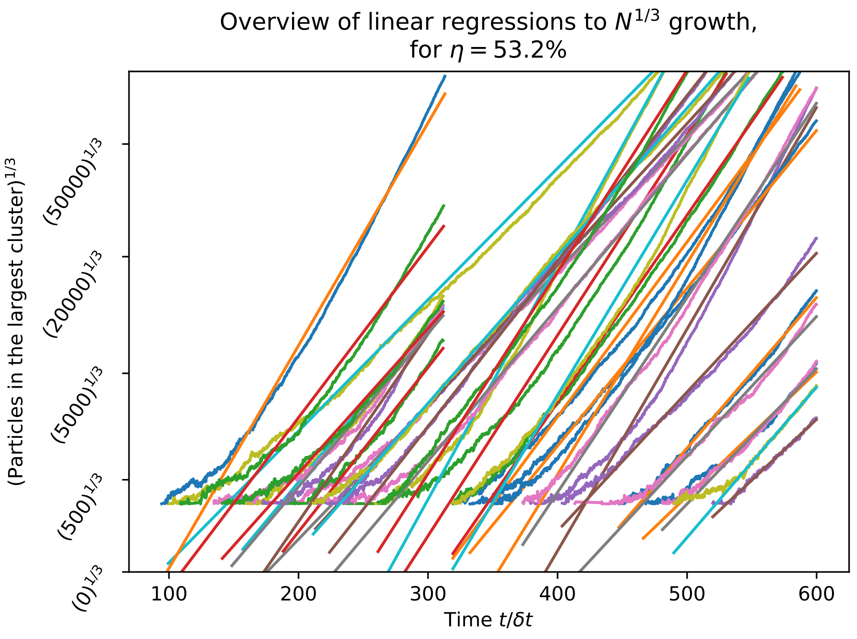
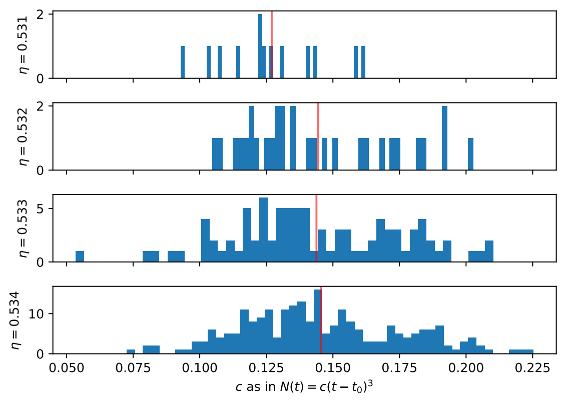
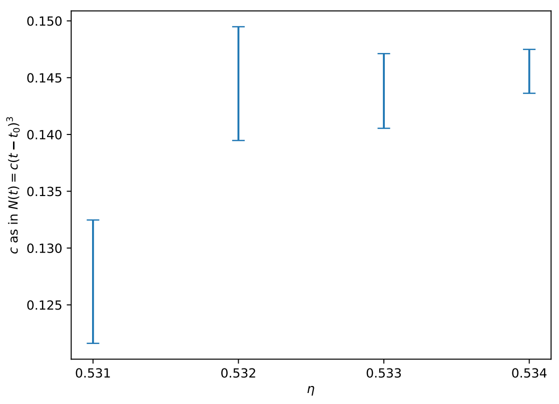


Figure 3.3.1: Trajectories of the third root of the number of particles within the largest cluster of a system over time. Clearly visible is the linear proportionality for which a linear regression is shown together with the data. The cut of some data sets at $t/\delta t \approx 300$ is due to the trepassing of the maximum walltime of the NEMO cluster. This means that the simulation of 20000 production steps yields a system time of $T/\delta t \approx 300$. Clusters present already in the first step would become to large in the next simulation intervall leading to a breach of the walltime limit due to the quadratic effort reuired for the q6q6 cluster finding routine. It can be assumed that clusters forming just around $t/\delta t \approx 300$ might not have been recognized due to this flaw. But as the number of trajectories concerning this is rather small the impact is not easy to reckognize when looking at the induction time distributions in **??**.

What we see from the histograms is that the distribution is rather spread out, but interestingly not signigicantly depending on the volume fraction. Except for $\eta = 0.531$ we find a smaller groth rate. A possible explanation for this behaviour could be that growth by heterogenous crystalization on the growing cluster surface, leading to a mean higher growth rate for higher volume fractions, is less likely for the lower volume fracitons. Either way due to the low statistics at the lowest volume fraction it is also possible that only a statistical fluctuation is seen. From the similarity of the growth rates we can deduce that the attachement of the particles to the cluster is a reaction controlled process. is this REACTION or diffusion controlled? And can we compare to literature in this case? In that case take up the densities



(a) Histograms of the slopes for the linear regressions to the largest clusters during the later constant growth process. The histograms are for $\eta = 0.531, 0.532, 0.533, 0.534$.



(b) Mean of the histograms with the uncertainty on the mean given by $\sigma_{\langle c \rangle} = \sigma_c / \sqrt{n}$ with n being the number of measurements included in the average. is the standard deviation here more sensefull, as interest lays on the widht of the distribution not so much on the mean?

Figure 3.3.2: Comparison of growth rates in the constant attachement regime.

As the diffusion constants vary from $D = 0.0081|_{\eta=0.532}$ to $D = 0.0075|_{\eta=0.534}$ they span a difference of about 7.5%, but does that mean they are either reaction or diffusion controlled, or is their only nothing to see, as the uncertainty on the growth rate is also of the size 5 % ?

3.4 Tensor of Gyration properties

The tensor of gyration is a very usefull tool as it describes the second moments of the position distributions. Such it comprises information about the spatial extent in all three dimensions, from which we can derive quantities as the radius of gyration, asphericity or anisotropy which will be defined following the definitions in the wikipedia either find an other source or cite it correctly.

The tensor of gyration is defined by eq. 3.4.1.

$$S_{mn} = \frac{1}{N} \sum_{i=1}^{N} r_m^{(i)} r_n^{(i)} \tag{3.4.1}$$

$$\text{with} \quad \sum_{i=1}^{N} \vec{r}^{(i)} = 0 \tag{3.4.2}$$

As described by eq. 3.4.2 the matrix S_{mn} is calculated in the center of mass frame for particles with the same mass. Furthermore the tensor of gyration can be diagonalized, with the three Eigenvalues

λ_1^2, λ_2^2 and λ_3^2. The cartesian system thereby is choosen such that $\lambda_1^2 \leq \lambda_2^2 \leq \lambda_3^2$. These Eigenvalues correspond to the spatial extents of the cluster within the cartesian system in which the tensor of gyration becomes diagonal. From these three Eigenvalues the quantities defined in eq. 3.4.3 - 3.4.6 are common to characterize clusters of particles.

$$\text{(squared) Radius of gyration:} \quad R_G^2 = \sum_{i=1}^{3} \lambda_i^2 \tag{3.4.3}$$

$$\text{Asphericity:} \quad b = \lambda_3^2 - \frac{1}{2}(\lambda_1^2 + \lambda_2^2) \tag{3.4.4}$$

$$\text{Acylindricity:} \quad c = \lambda_2^2 - \lambda_1^2 \tag{3.4.5}$$

$$\text{Relative shape anisotropy:} \quad \kappa^2 = \frac{b^2 + \frac{3}{4}c^2}{R_G^4} = \frac{3}{2}\frac{\sum_{i=1}^3 \lambda_i^4}{\left(\sum_{i=j}^3 \lambda_j^2\right)^2} - \frac{1}{2} \tag{3.4.6}$$

For better understanding of the shape descriptors mentioned before, we can have a more detailed look at their interpretation:

Radius of gyration R_G:

An averaged radius of the structure.

Asphericity b:

The difference of the largest extrent with an average of the two smaller extents.

Acylindiricty c:

The difference of the smaller extents

Relative shape anisotropy κ^2:

A sum of the asphericity and the acylindricity normalized by the radius of gyration to obtain a dimensionless quantitiy between 0 and 1.

To spot possible correlations between cluster shape and growth rates at early or late stages as well as correlations concering the induction time the three quantities eq. 3.4.3 - 3.4.6 derived from the tensor of gyration have been first plotted versus the cluster size when the quantity was measured to make them comparable. As the number of particles in the cluster is not monotonously increasing, the depicted quantity is not a function anymore. Either way it makes sense to show the data in this form, as we expect that we can compare clusters best at a defined number of particles. Furthermore the number of particles, as well as the quantities can span many orders of magnitude making logarithmic scales useful.
To finally highlight correlations between the aforementiond growth related scalar quantities and the tensor of gyration quantities, the single trajectories have been coloured by the corresponding magnitude

of the scalar quantity. A large overview produced by this procedure is given in fig.3.4.1 for the nucleated trajectories at $\eta = 0.534$.

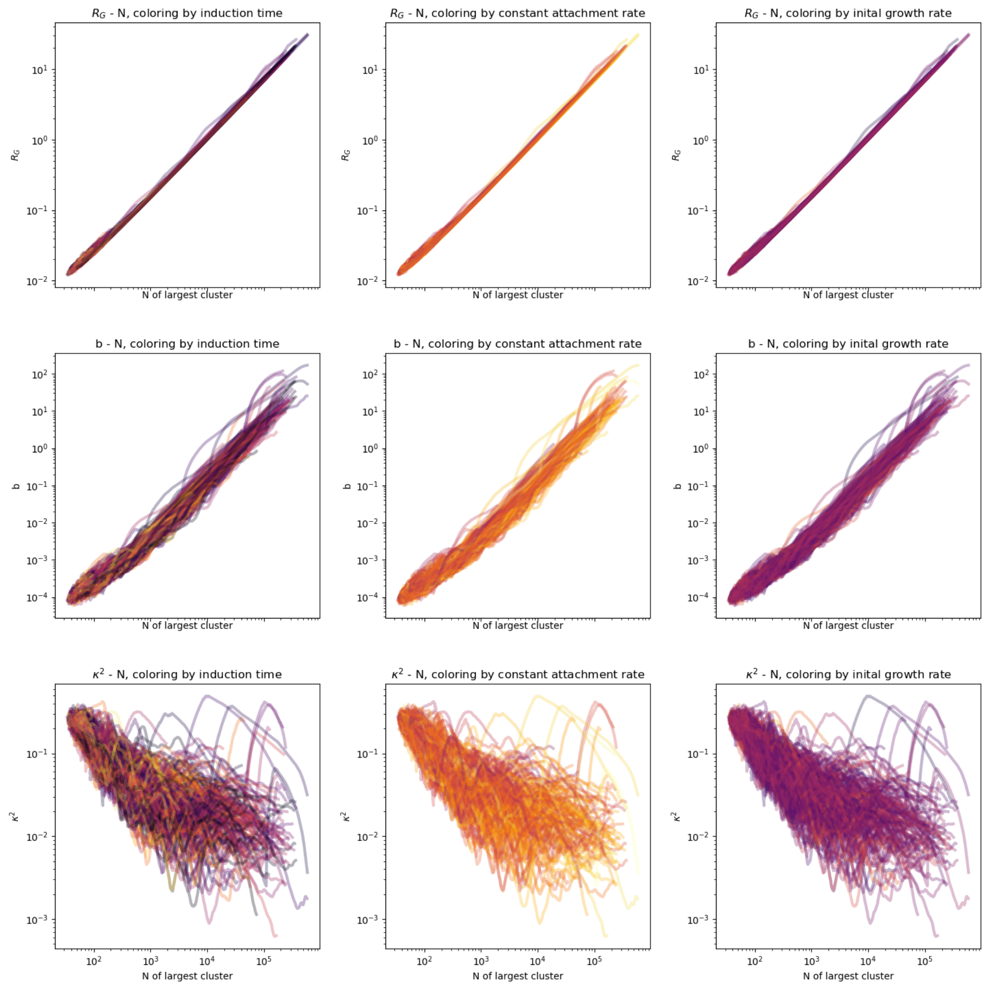


Figure 3.4.1: Overview of the cluster shape describing quantites radius of gyration (R_G), asphericity (b) and anisotropy (κ^2), depending on the size of the cluster. The colouring depicts the scalar quantities induction time, constant attachment rate and inital growth rate.

With the overview it was tried to capture different characeristic features of the cluster growth. The induction time is choosen, because clusters nucleating shortly after the quench might start with a less organized structure which could lead to a higher asphericity at early times.

Also a possible idea would be that clusters starting with a slow growth at the beginning might be less structured, as this could make attachement of new particles harder and melting more likely. For quantifying the inital growth rate, an exponential function has been fitted to the data up to a cluster size of 500 particles. The growth rate within the exponential function is taken as measure on how swift the nucleation process works from the precursor to the growing crystal.

The third scalar quantity is the attachement rate of the cluster at later times. As this attachment rate might be linked to the number of defects within the crystal or other structural properties like the present crystal strucure or purity of the crystal it was also thought to be corrolated to easily accesible properties of the cluster like the shape descriptors.

But from the overview we get no obvious sign that there are any correlations between cluster shape and growth rates or between cluster shape and the induction time. Because of that no deeper analysis is done, but instead we conclude that by this superficial analysis we cannot relate the shape descriptors of the cluster to growth or structural properties.

3.5 Autocovariance functions (largest cluster, p(n,t))

talk about if this is usefule in any way.

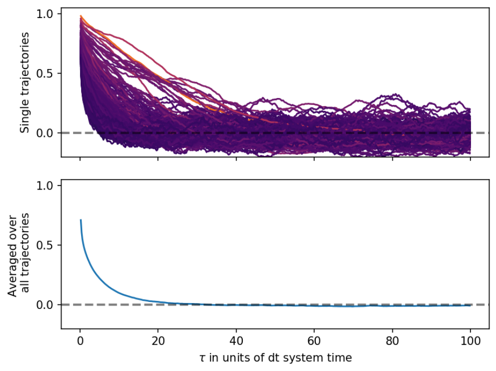
The autocovariancefunction (ACF) of the largest cluster contains information about how long a single cluster persists as the largest cluster within the volume, as flucutations of clusters at different points of the volume are expected to be independent of each other, the size fluctuation of a distinct cluster should be corroloated in time . (Except if we think about the modes within the fluid encompassing not only local areas...)

The autocovariance function is defined by eq. 3.5.1, where $N_{lc}(t)$ is the number of particles in the largest cluster at time t, $\langle N_{lc} \rangle_t$ is the corresponding average over time, thus $X(t)$ describes the deviations from the average. The autocovariance function furthermore is normalized by $\langle X^2 \rangle$ which can be identified as the variance of the data. With this normalization $ACF(\tau = 0) = 1$.

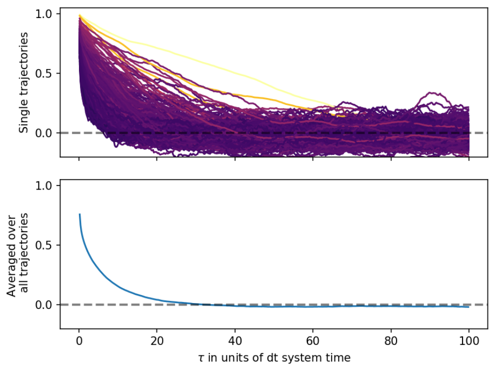
$$ACF(\tau) = \frac{\langle X(\tau) - X(0) \rangle}{\langle X^2 \rangle} \tag{3.5.1}$$

$$\text{with } X(t) = N_{lc}(t) - \langle N_{lc} \rangle_t \tag{3.5.2}$$

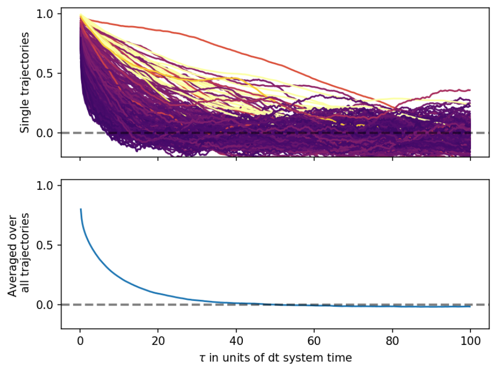
The calculation is performed on the time resolved largest cluster within a measurement. As the ACF is supposed to give insight to the fluctuations of clusters in the metastable fluid only measurements that did not nucleate or at least involved only little cluster growth were used for the ACF in fig. 3.5.1. From the autocovariance functions we see that stuructural flucutation seen as clusters persist for longer times at higher volume fractions. From the colorring as well as from the cluster distributions we can also conclude that the fluctuations tend to be larger at higher volume fracitons.



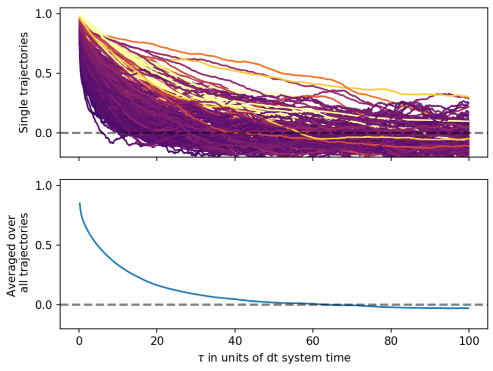
(a) $\eta = 0.531$



(b) $\eta = 0.532$



(c) $\eta = 0.533$



(d) $\eta = 0.534$

Figure 3.5.1: Comparison of autocovariance functions in the metastable fluid. The colouring corresponds to the largest cluster present over the whole simulation time. The lightest colour indicate therby a largest cluster of more than 500 hundred particles which can more or less also be called nucleated, but these are rare in the given selection and such the data represents the metastable fluid still rather good.

3.6 Nuclection time dilemma

To later on calculated induction times or average nucleation times, we will require a definition of when a crystal is called nucleated. This means we have to define from which point a cluster is not merely a unstable fluctuation within the liquid anymore, but instead becomes a stable solid crystaline phase. In the literature many concepts are used. For example a cluster can be defined as crystaline soon as its of the critical size, calculated by CNT or by doing a comitter analysis. An other possibility often used is to rewind the trajectory soon as a clearly stable crystal is found, up to the point when the crystal clusters size more or less vanishes. A further approach is to fit the growth during later times and extrapolate it to the time when the cluster vanishes.

All these definitions differ more or less only by a delay Δ_τ which is a distribution of times holding the information of how long it takes for varying clusters to pass from the first criterion to the next.
For example we can take as a first point the time when a cluster, known to crystalize at later times, cannot be differentiated anymore from any other structural fluctuation in the liquid, i.e. when the size of the cluster falls below some threshold given by the size of clusters regularly present in a given volume.
The second point we can set by either the critical size of CNT or by some other criterion when we are sure that the cluster has stabilized and will only continue to grow.
At the first of these two points, the fluctuation leading to the crystalization occurs but it would not be possible to tell yet if this precursor melts or continues to grow, while at the second point the crystal is stable. Such the first might be called a precursor nucleation and the second crystal nucleation. Between these two points we find the time difference to be the time it takes for the precursors to form a stable crystal. This includes also that some precursors might loiter for awhile before forming the stable phase while other pass this gap rather directly.

When calculating a mean induction time, the delay Δ_τ propagates also to the final result and as it is a stochastic distribution also its higher moments are propagated leading to a smaller precision. This afterall only means that the induction time depends on the definition of crystalization and they are only roughly comparable.
In fig. 3.6.1 three distirbutions with varying definitions of the induction time are visualized.
The three methods explicitly used here are given by the following:

Horizon crossing The time of nucleation is obtained by following the trajctory of the largest cluster within a system after it clearly nucleated back to the point where it was the last time at the average largest cluster of the metastable fluid without stable cluters. The name horizon crossing refers therby to the idea that fluctuations of the largest cluster in the metastable fluid are more or less independent fluctuations. This comes by the fact that the large fluctuations of the system alternate at being the largest one, and such the largest fluctuations is not bound locally and the fluctuations becomes independent events. On the other side an extraordinary fluctuation will be seen for a longer period of time as the largest cluster, and thus the corresponding flucutuations are not independent in time. The crossing of the trajectory below this horizon where it cannot be followed any longer is meant by the name.

Exponential extrapolation For this method an exponential growth is fitted to the largest cluster data up to N<500. Extrapolating to smaller times makes it possible to evaluate when the exponential crossed 10 particles, which is then taken as the induciton time. The method tends to find negative induction times that are not physical.

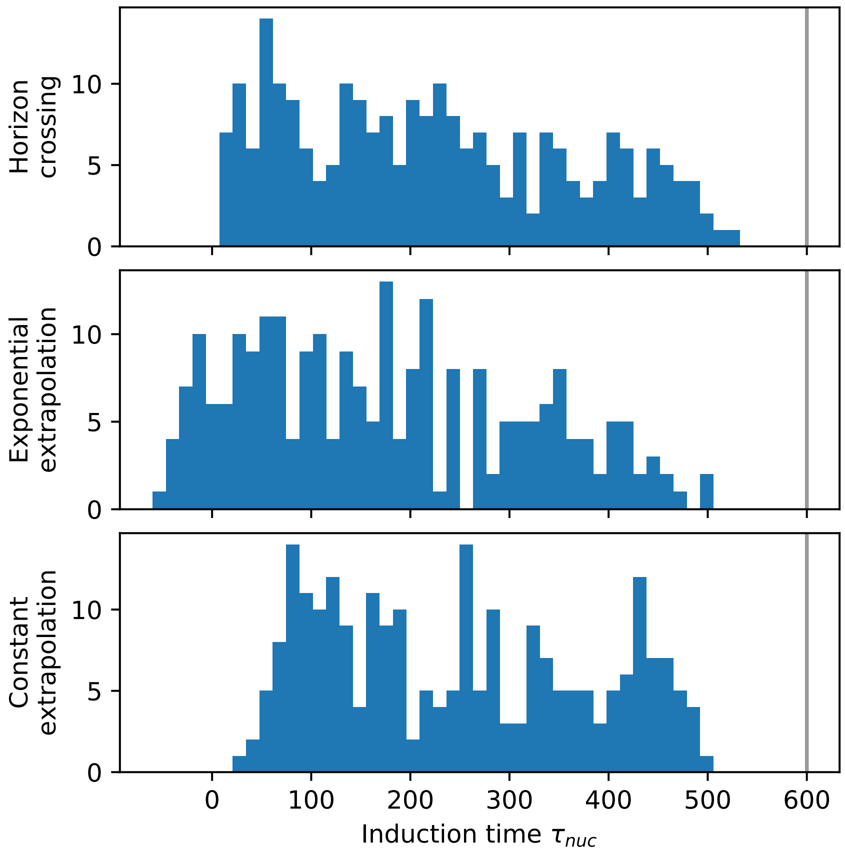


Figure 3.6.1: Induction time distribution obtained by different definitions. While the two methods using extrapolation seem to have the two effects of smearing the signal as well as shifting them, the method of defining the nucleation as the time when the largest cluster is last below the horizon of fluctuations seems to return the most accurate and precise distribution. The final simualtion time is mrked by the grey line. As clusters require some time to be clearly recognised as crystals no nucleation are seen towards the end of the simualtion intervall. To counteract this we will truncate the distribution in the following analysis such that this does not have an impact on the final result.

Constant extrapolation The name refers to the constant attachment rate found at later times for the cluster growth. It can be extrpolated to earlier times until the cluster has completly vanished i.e N=0. As the constant attachement rate is higher than the inital growth rate this method returns too large induction times.

As can be seen the horizon crossing method returns a rather smooth distribution that also roughly can be approximated by an exponential decay that is expected for a constant nucleation rate as will be shown in put in the correct section or equation.

3.7 Induction time by exponential distribution

Estimating the induction time of the hard sphere nucleation has been done over the previous two decades multiple times by experimentalists and theorists. The employed procedures vary do they vary or are they just the mean every time? Well the experimentalists use the point when the whole sample crystalizes. between experimentalists and theorists as the experimentalists often have access to very large systems but wihtout knowing all positions at all times, while the theorists mostly have smaller system in simulations with the adavantage of being able to access all particle positions and in case of simulations probing the dynamics also at all times.

Furthermore theorists have used simple approaches like the average induction time which has involves the constraint that all trajectories of an ensemble are required to have nucleated. This is very suitable and simple for systems at large volume fractions as for them all simulations of an ensemble might be calculated up to nucleation. At lower volume fractions this becomes more and more unfeasible due to the steep increase of the induction time.

To cirumvent this problem we will define the nucleation rate in the following differently without requiring all simualtions to nucleate. In fact we can also show that the uncertainty of the induction time obatained from the data is not signifiantly reduced anymore for measurements longer than the induction time.

3.7.1 CNT expectation the induction time distribution

In section 1.3 we introduced Classical nucleation theory and its constant nucleation rate depending on the barrier height in the free energy landscape. Even if there are signs that CNT is not appropriate for describing nucleation process perfectly, we will use its prediction of a constant nucleation rate as an assumption to define such a constant scalar nucleation rate.

As also mentioned before, in the discussion of the system sizes (section 3.1), the induction time of a system depends on the volume under consideration and such the nucleation rate is commonly defined as a nucleation rate density k.
Considering now the simualtions we can describe them as a total of m volumes with a size of V_{box}. Further we can define the number of boxes in which a nucleation occured as $n(t)$ and exclude these from the further simulation.

In this case the total nucleation rate $\dot{n}$ can be written by eq. 3.7.1 from which in the continous limit of an infinte number of different simulations we can deduce the expected induction rate.

$$\dot{n} = (m - n(t))V_{box}k \tag{3.7.1}$$

$$\Leftrightarrow \frac{\dot{n}}{m} = (1 - \frac{n(t)}{m})V_{box}k \tag{3.7.2}$$

$$\text{in the limit } m \to \infty$$

$$\Leftrightarrow \frac{n(t)}{m} = 1 - \exp\left(-V_{box}kt\right) \tag{3.7.3}$$

$$\text{defining } \tau = (V_{box}k)^{-1}$$

$$\Leftrightarrow \frac{\dot{n}(t)}{m} = \frac{1}{\tau}\exp\left(\frac{-t}{\tau}\right) \tag{3.7.4}$$

The final result in eq. 3.7.4 is the well known stochastic exponential distribution. As the expectation value of the exponential distribution is given by its parameter τ the common approach of using the mean induction time when all simulations have nucleated yields an accuracte result and precision can be obtained by taking a large number of simulations.

3.7.2 Maximum likely estimator of induction time

In case the simulation time is not feasible we instead will have to deal with truncated exponential distributions. For this we can use Maximum likelihood estimators. The derivation follows cite the statistics paper.

Maximum likelihood estimators are based on the idea that we can write down the expression of the total probability called likelihood $\mathcal{L}$ for a given set of measurements x_i depending on parameters of the assumed underlying distribution. For the exponential distribution parametrized by the characteristic decay rate κ this is given by eq. 3.7.5.

$$\mathcal{L}(\kappa) = \prod_{i=1}^{N} p(x_i) = \prod_{i=1}^{N} \kappa^N \exp\left(-\kappa x_i\right) \tag{3.7.5}$$

During the process we try to find the maximum of this product. To simplify this product and also to evade overflow problems on floating point machines, the logarithm of the likelihood is used and maximized yielding the same parameters because the logarithm is a monotonic function and thus does not shift the extrema.
The maximum probability can then be found by usual means of analysis executed in eq. 3.7.6.

$$0 \overset{!}{=} \left.\frac{\partial \log(\mathcal{L})}{\partial \kappa}\right|_{\kappa=\hat{\kappa}} \tag{3.7.6}$$

$$\Leftrightarrow \quad 0 = \frac{\partial}{\partial \kappa}\left.\left(N\log(\kappa) - \kappa\sum_{i=1}^{N} t_i\right)\right|_{\kappa=\hat{\kappa}} \tag{3.7.7}$$

$$\Leftrightarrow \quad 0 = \frac{N}{\hat{\kappa}} - \sum_{i=1}^{N} t_i \tag{3.7.8}$$

$$\Leftrightarrow \quad \hat{\kappa}^{-1} = \frac{1}{N}\sum_{i=1}^{N} t_i \tag{3.7.9}$$

Such we have found that the maximum likelihood estimator of κ for a set of samples drawn from an exponential distribution is given by the inverse arithmetic mean of the samples. This result is neither new nor surprising but is shown to illustrate how the method of maximum likelihood works. In the following we then show how to handle censored and truncated distribtuions by the maximum likelihood method.

Both terms in this context refer to sets of samples that are incomplete in the sense that they only include samples up to some threshold $t_i < T$. In the case of truncated distributions the number of samples larger than this threshold is unknown while for the censored distribution the number of samples is known. Taking the example of time consuming nucleation events in computer simulations we are in the case of censored distributions, as the total number of simulation boxes is known but the simulation is stopped at some point when enough nucleations have been collected and such the number of samples that would have nucleated at later times is known. The probability of an event after the end of the simulation is given by eq. 3.7.10.

$$p(t_i > T) = \int_T^\infty \kappa\exp(-\kappa t)dt = \exp(-\kappa t) \tag{3.7.10}$$

The probability distribution not only below the threshold but also above can then be written as in eq. 3.7.11.

$$f(t) = \begin{cases} \kappa\exp(-\kappa t) & t < T \\ \exp(-\kappa T) & t \geq T \end{cases} \tag{3.7.11}$$

In the simulation we can split up the number of boxes N, into n boxes where a nucleation event was found, and $m = N - n$ others where no nucleation event was spotted during the simulation time T. Further we have to account for the fact that the samples without distinct times are indistinguishable. This is done by weighting them with the number of possible permuations given by the binomial prefactor $\binom{N}{m}$. The whole expression is then given in eq. 3.7.12 and the extremum of the likelihood function is evaluated in the subsequent reformulation.

$$\mathcal{L}(\kappa) = \binom{N}{m} \kappa^n \ \exp(-\kappa \sum_{i=1}^{n} t_i) \ \exp(-\kappa T)^m \quad \Bigg| \frac{\partial \log(...)}{\partial \kappa}\Bigg|_{\kappa=\hat{\kappa}} \tag{3.7.12}$$

$$\Leftrightarrow \quad \log(\mathcal{L}(\kappa)) = \log \binom{N}{m} + n \log(\kappa) - \kappa \sum_{i=1}^{n} t_i - m\kappa T \quad \Bigg| \frac{\partial (...)}{\partial \kappa}\Bigg|_{\kappa=\hat{\kappa}} \tag{3.7.13}$$

$$\Leftrightarrow \quad \frac{\partial \log(\mathcal{L}(\kappa))}{\partial \kappa} = \frac{n}{\kappa} - \sum_{i=1}^{n} t_i - mT \quad \Bigg|_{\kappa=\hat{\kappa}} \tag{3.7.14}$$

$$\Bigg| \quad \text{with } \frac{\partial \log(\mathcal{L}(\hat{\kappa}))}{\partial \kappa} \overset{!}{=} 0$$

$$\Leftrightarrow \quad 0 = \frac{n}{\hat{\kappa}} - \sum_{i=1}^{n} t_i - mT \tag{3.7.15}$$

$$\Leftrightarrow \quad \hat{\kappa}^{-1} = \frac{1}{n} \left(\sum_{i=1}^{n} t_i + mT \right) \tag{3.7.16}$$

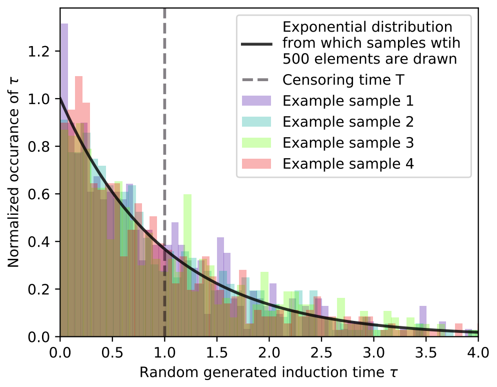
The final line eq. 3.7.16 is the estimator of the decay rate of the censored exponential distribution. It is used for the estimation of induction times to compare with other published results in the next sections.

3.7.3 Monte Carlo uncertainty estimaion

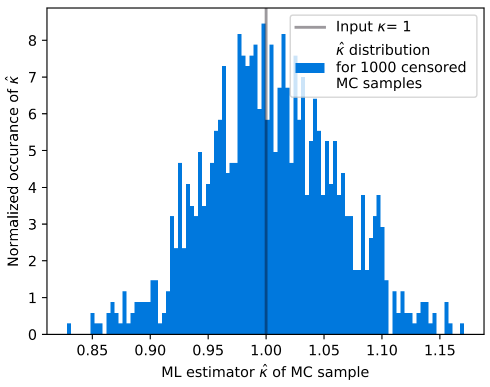
Having found the estimator the next question is what is its uncertainty, i.e what is the distribution of κ. Eventhough corresponding literatur on analytic expresssions of the distribution exist the complexity becomes inappropriate for the task at hand. We will follow instead a Monte Carlo method proposed for example in the book Numerical Recipes cite numerical recispes to find the uncertainty of the estimator.

For this purpose we draw samples from an exponential distribution characterized by the estimator calculated from the actual simualtion data. Afterwards the samples are censored by cutting off all elements larger than T and calculate the corresponding estimator $\hat{\kappa}_{MC}$ for the Monte Carlo sample. From multiple such random sets we can create a histogram of estimates for $\hat{\kappa}$ that can be seen together with some exemplary random samples in fig. 3.7.1. As the distribution seems to incorporate only little higher moments the standard deviation of the distribution is used as the uncertainty $\sigma_{\hat{\kappa}}$.

Concerning the uncertainty in detail we can ask how long a simulation should be to yield precise results. For this we can first look at the case where $1 \gg \kappa T$ corresponding to a simulation where all boxes showed an nucleation event. In this case we have seen before that $\hat{kappa}^{-1} = \frac{1}{N} \sum_{i=1}^{N} t_i$. As we assume that the t_i are exponentially distributed we further know that $\sigma_t = \kappa^{-1}$. The gaussian error



(a) Exponentially distributed random samples of size 500 with an exemplary censoring time of $T = \kappa^{-1}$



(b) Distribution of $\hat{\kappa}$ for the previously generated MC samples. The distribution can be described mostly by mean and standard deviation as the number of estimates in the tails are small.

Figure 3.7.1: Exemplary samples for a given κ as well as the distribution of estimates calculated from the random samples. The uncertainty on $\hat{\kappa}$ is approximated by the standard deviation of the distribution from the corresponding Monte Carlo analysis at a given κ.

propagation then is given in eq. 3.7.17.

$$\frac{\sigma_\kappa}{\kappa} = \frac{1}{\kappa}\sqrt{\sum_{i=1}^{N}\left(\frac{\partial\kappa}{\partial t_i}\right)^2 \sigma_{t_i}^2} \tag{3.7.17}$$

$$\left|\begin{aligned} \text{with } \frac{\partial\kappa}{\partial t_i} &= \frac{\partial}{\partial t_i}\left(N\left(\sum_{i=1}^{N} t_i\right)^{-1}\right) \\ &= -N\left(\sum_{i=1}^{N} t_i\right)^{-2} = \frac{\kappa^2}{N}\,, \\ \text{and } \sigma_t &= \kappa^{-1} \end{aligned}\right.$$

$$= \frac{1}{\kappa}\sqrt{N\left(\frac{\kappa^2}{N}\right)^2 \kappa^{-2}} \tag{3.7.18}$$

$$= \frac{1}{\sqrt{N}} \tag{3.7.19}$$

Is das eine Tautologie!?

Similarly we can take the limit of $1 \ll \kappa T$ which is the case when the mean nucleation time is much larger than the simulation time and such only a small fraction of the boxes hosted a nucleation event. In this case we can expand the estimator in the fraction of nucleated trajectories $\frac{n}{N}$ to find $\hat{\kappa} \approx \frac{n}{N}\frac{1}{T}$.

In this case the decrease of nucleations events due to a smaller amount of available totoal volume is not seen yet, and the only information about the nucleation rate is obtained from the number of boxes with nucleations compared to the number of total amount of boxes used. As n is poission distributed we know that $\sigma_n = \sqrt{n}$. Fixing N and T and using the expectation value of nucleations $n = N\kappa T$ the gaussian error propagation for the relative uncertainty is given in eq. 3.7.20.

$$\frac{\sigma_\kappa}{\kappa} = \frac{1}{\kappa}\frac{\sqrt{n}}{NT} \tag{3.7.20}$$

$$= \frac{\sqrt{N\kappa T}}{NT\kappa} \tag{3.7.21}$$

$$= \frac{1}{\sqrt{N\kappa T}} \tag{3.7.22}$$

Finally we are also able to not only look at limits analytically, but also to approximate the relative uncertainty directly by means of the aforementioned Monte Carlo method. For this purpose the same procedure as before is used. The number of elements per sample is consistently with the performed simulations taken to be 500 and to archieve good precision the standard deviation of 1000 samples is used for the uncertainty. As can be seen in fig. 3.7.2 the fluctuations between different evaluations becomes rather small, but increase if using a lower number of samples. To compare the analytically derived limits of the uncertainty with the Monte Carlo results both are drawn into fig. 3.7.2.

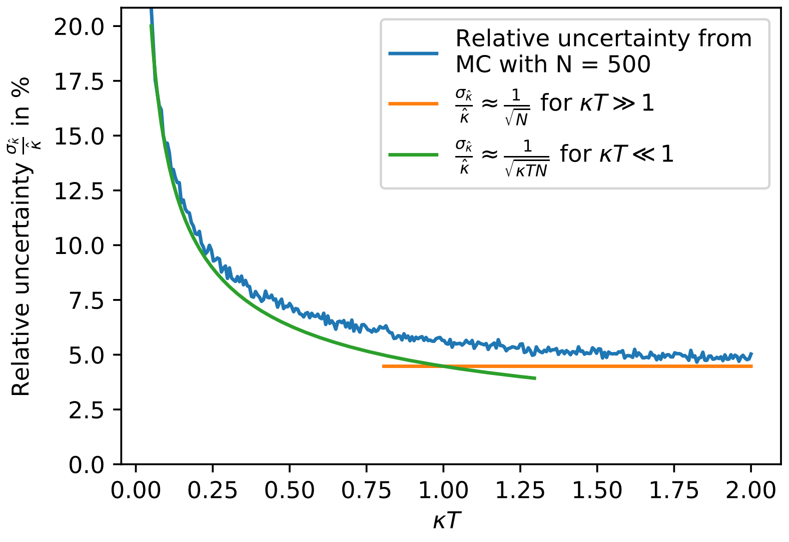


Figure 3.7.2: Relative uncertainty of the ML estimator for varying κT. The x scale is choosen dimensionless such that it indicates the simulation time in comparison to the characteristic nucleation time.

We find that for the limits of $\kappa T \ll 1$ as well as $\kappa T \gg 1$ Monte Carlo results and analytical results are in good accordance while in between the analytical limits only can be used as a rough estimate.

What can be seen from fig. 3.7.2 is that the uncertainty of the estimation drops sharply until about half of the characteristic time, after which it only gains little more precision. This is not surprising as the information is contained in the nucleation times and rather fast many nucleations have occured and the long simulation times add only little of further nucleations. Thus simulating until all boxes had an nucleation event is only necessary if one want s to use the simpler arithmetic mean of the induction times as a characteristic time, or if any oher constraints make it necessary to reach nucleation of all boxes.

3.8 Nucleation rate comparison

Finally we are able to evaluate the induction time distribution to find
All Nucleation rates that can be found.-> mayhap ask Hajo.
-nucleation rates without
-nucleation rates with small particles

3.9 Memory Kernels

Memory kernels of systems at various densities. Depends strongly on what is found here
-memory kernels of 16k system at varying points of time
-memory kernels from fall article
-maybe memory kernels of 1m system, but do not know what to say. Maybe only mention that after seeing the 16k system to have only memory kernes at about middle of the time, no true memory kernel is visible in the 1m system as volume fractions have been to low with to large calculation times.

4 Conclusion - Summary

4.1 Conclusion

5 Appendix

.1 A