Computational Science

Molecule Dynamic simulation

Task:

Implementation of an MD-simulation for Lennard-Jones particles/ ions/ water in 2D and 3D. →interaction between the particles an thereby causes spatial motion by iterativ calculations

Computational Science 1

- Task: 1
- Concept: 3
- Workflow 4
- So, 13.01.19 4
- Selbstständige Arbeitsphase 4
- Sa, 19.01.19 4
- Arbeitsphase 4
- So, 27.01.19 5
- Mo, 04.01.19 5
- Arbeitsphase 5

Any: 5

Phillip : 5

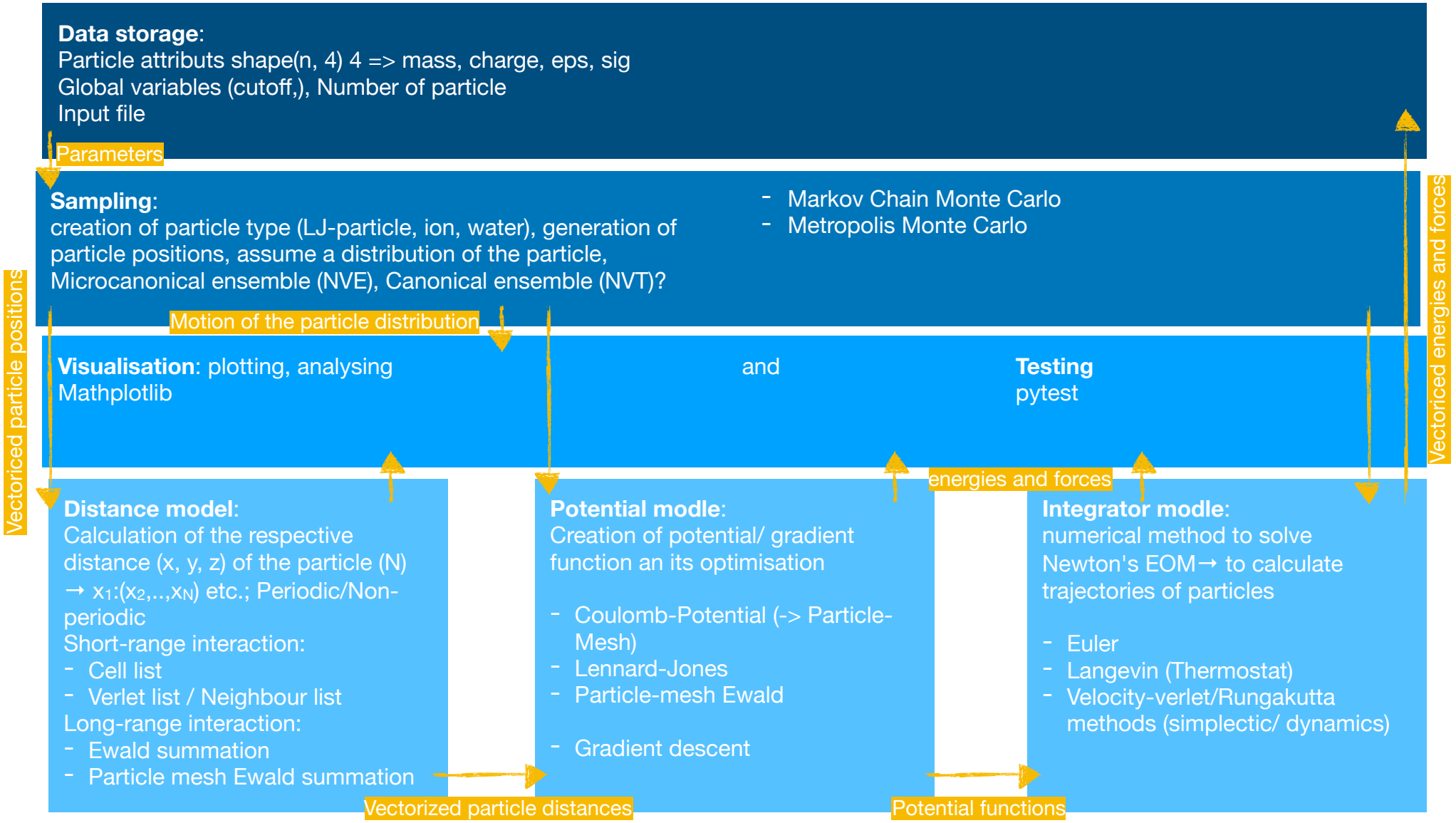
Felix: 5

Malte: 5

- Next steps 6
- Possible next steps 6

Overview: 7
Information about modules 9
Data storage module 9
Sampling module 9
Distance module 10
Potential module 11
Integrator module 11
Minimisation 11
Lecture notes 12
Präsentation 13

Concept:


Data storage:
Particle attributs shape(n, 4) 4 => mass, charge, eps, sig
Global variables (cutoff,), Number of particle
Input file
Parameters
Sampling:
creation of particle type (LJ-particle, ion, water), generation of particle positions, assume a distribution of the particle, Microcanonical ensemble (NVE), Canonical ensemble (NVT)?
- Markov Chain Monte Carlo
- Metropolis Monte Carlo
Vectoriced particle positions
Motion of the particle distribution
Vectoriced energies and forces
Visualisation: plotting, analysing
Mathplotlib
and
Testing
pytest
energies and forces
Distance model:
Calculation of the respective distance (x, y, z) of the particle (N) → x_1:(x_2,..,x_N) etc.; Periodic/Non-periodic
Short-range interaction:
- Cell list
- Verlet list / Neighbour list
Long-range interaction:
- Ewald summation
- Particle mesh Ewald summation
Potential modle:
Creation of potential/ gradient function an its optimisation
- Coulomb-Potential (-> Particle-Mesh)
- Lennard-Jones
- Particle-mesh Ewald
- Gradient descent
Integrator modle:
numerical method to solve Newton's EOM→ to calculate trajectories of particles
- Euler
- Langevin (Thermostat)
- Velocity-verlet/Rungakutta methods (simplectic/ dynamics)
Vectorized particle distances
Potential functions

Workflow

So, 13.01.19

✓ finale aufgaben Verteilung
✓ modules diskutieren
- Labeling festlegen (formate, Bezeichnungen, Abkürzungen, Parameter)
✓ Input und Output formate und Benennungen definieren
✓ Ziele festlegen für Fortschritt bis zum nächsten Treffen
- Parallelization strategies ? Domain decomposition method
- (Distribution of system data for parallel computing

Selbstständige Arbeitsphase

✓ Module fertig implementieren

Sa, 19.01.19

✓ Fortschritte präsentieren
✓ Fehler Probleme diskutieren
- testing diskutieren
- Input Beispiele
- Labeling festlegen (formate, Bezeichnungen, Abkürzungen, Parameter)
✓ Erste Ideen Wie schreiben wir alles zusammen
- Parallelization strategies ? Domain decomposition method
✓ Erster primitiver Testlauf

Arbeitsphase

- Visualisierung, testing.
- Sampling optimieren (diffrend properties for particles?)
- Integrator timing , visualisation closed-, pbc- system, grid ?
✓ Minimazation
✓ Ewald- summation
✓ Create classes and Bild the structure of the hole code

So, 27.01.19

✓ input examples
✓ Combine all modules
- Discuss extra features
    - Water

Mo, 04.01.19

✓ presentation ewald
✓ Presentation optimiser
✓ Define the next steps

Arbeitsphase

ANY:
- integrator + box condition, Temperaturerhöhung in Langevin: heat up or down, (Donnerstag)
- Visualisation
    - Kin & pot energy
    - Energy consistency
    - histogram
    - Benchmark potential ?
- piecewise function for LJ-cutoff ? (ab 22.02)

PHILLIP :
- timing moduls,
- metropolis hasting,
- Maxwell-Boltzmann Anfangsgeschwindigkeit je nach Temperatur,
- simulation test

FELIX:
- Weiter ab 14.02

MALTE:
- gradients (Hesse)

Next steps

- Units in plotter module
- Integrator 1000 particle 100 steps 30 sek?
- Code optimisation
- Documentation
- Sampler
  - Different input examples
  - Different properties for particles?
- Post simulation analysis
  - Mean distance
  - ??
- For each module for presentation
  - Appropriate test
  - Visualisation
  - Code overview

Possible next steps

- Water bath temperature control
- User surface
- Discuss extra features
    - Water
    - working with grid —>particle.mesh Ewald
- Implementing grid ?
- dara storage ?
- codacy : https://app.codacy.com/login und das andere ?
-

Overview:

Aufgabe für (Person): Phillip (P), Felix (F), Malte (M), Joana (J)

MODUL	INPUT	TASK	OUTPUT	TESTING	VISUALISATION
DATA STORAGE		Configuration ?			
POTENTIAL (F)	C: - Distance shape(nxn), - charge array(n), - ε, σ(nx1), forces LJ: - dis(nxn), - ε, σ(nx1) H: - Box dim (x,y,z min and x,y,z max), - Positionen: r (einteilchen or Mehrteilchen potential), r_0= xmax/ 2, k ???	<u>Particle-particle:</u> - Coulomb - LJ <u>Global potential:</u> - Harmonic	C: - potential(nx1) or (Gesamtpotential for each particle) - gradient shape(nx3) LJ: - potential(nx1) H: Potential	- energy consistency - Use simple potential with work and compare	
SAMPLING (P)	- N = number of particle - for each particle charge, mass, ε, σ (nx1), k (federkonstnat) ‣ Creating a particle ‣ box, pbc, non pbc GLOBAL atrributs: ???	configuration: Random then: - Markov Chain Monte Carlo - Metropolis Monte Carlo	- Position shape(nxdim) - Velocity shape(nxdim)	-	- distribution of particle plotten -

MODUL	INPUT	TASK	OUTPUT	TESTING	VISUALISATION
DISTANCE (P)	- Position shape(nxdim)	➡ box, pbc, non pbc	- dist(nxn)		
INTEGRATOR (J)	- position shape(nxdim), - velocity shape(nxdim), - attribute shape(n, #numb of attributes) - gradient shape(n,dim), - Time step (int) - Total time (int) - Open or close system ‣ box, pbc, non pbc - global attributes ?	Euler Velocity Verlet Langevin	- position(size, n, dim) - velocity(size, n, dim) - Accerelation(size, n, dim)		
Optimization (M)	??	- gradient descent, - conjugate gradient, - Newton methods			

Information about modules

DATA STORAGE MODULE
????

SAMPLING MODULE
Initial configuration (markow chain, Particles equispaced in a box; 2D/ 3D)

MONTE CARLO

INPUT	OUTPUT	
initial configuration r(0), potential, size, beta, step	Position r, potential(r)	

(1) Start with initial configuration $\mathbf{r}^{(0)}$ and set k = 0.

(2) Fork=1,....,K:

(a) Sample random vector $\eta \in \mathbb{R}^N$ with $\eta_i \sim N(0,\sigma^2)$, i.e. an isotropic multivariate Gaussian distribution/ or function with variances σ^2.

(b) Propose new configuration $\mathbf{r}' = \mathbf{r}^{(k)} + \eta$.

(c) Accept new configuration with probability $p_{\text{acc}} = \min\left\{1, \exp\left(\frac{\phi(\mathrm{r}') - \phi(\mathrm{r}^k)}{k_B T}\right)\right\}$

If accepted, set $\mathbf{r}^{(k+1)} = \mathbf{r}'$, otherwise $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)}$.

Markov Chain Monte Carlo
- create samples from a possibly multi-dimensional continuous random variable, with probability density proportional to a known function.
- these samples can be used to evaluate an integral over that variable, as its expected value or variance.
- in Markov chain Monte Carlo methods are autocorrelated.
- a equilibrium distribution which is proportional to the function given.

Metropolis Monte Carlo
- Konfigurationen gemäß ihrer Boltzmann-Wahrscheinlichkeit + einfachen arithmetischen Durchschnitt
- zuerst eine zufällige Bewegung ausgeführt
- dann die Boltzmann-Wahrscheinlichkeit einer solchen Bewegung ausgewertet wird und die Wahrscheinlichkeit mit einer Zufallszahl verglichen wird.
- Ist die Boltzmann-Wahrscheinlichkeit der Bewegung größer als die Zufallszahl, wird die Bewegung akzeptiert. Andernfalls wird das System auf seine ursprüngliche Konfiguration zurückgesetzt.

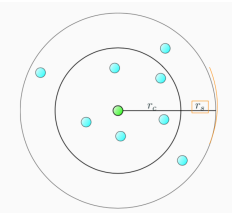
Vor- und Nachteile: ?

- not interested in the statistics of momenta anymore, so we integrate them out
- sampling the probability density.
- simple version of Monte Carlo is Metropolis Monte Carlo

DISTANCE MODULE

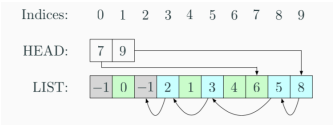
INPUT	OUTPUT	
Position r, potential(r)	Matrix distances x1 x2 … xN	

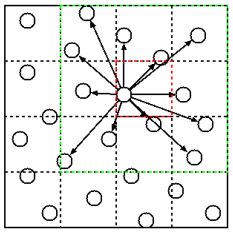
- calculation of distances between particles in MD Energy and force Calculation
- Introduce a cutoff and considering only the nearest neighbours within the radius; for LJ.Potential rc= 2.5 σ
  - Reduce effort: loop over all pairs of particle (i,j) = (j,i)
  -



Verlet list
- For each particle i, a Verlet list hat lists all other particles j within a smaller distance than rc+ rs
- Update the neighbour list every Ns tilmestep
- $r_s > N_s v_{\text{typ}} \delta t$ type speed of particles
- Choice between hash table and array

Cell-linked list : get complexity down
- domain into cells with an edge length greater than or equal to the cut-off radius of the interaction to be computed
- particles sorted into cells and the interactions are computed between particles in the same or neighbouring cells
- Needed array HEAD and array LIST
- Think about parallelisation





Cell decomposition approach
- partition the domain into cells and build a cell-linked list
- Verlet list by searching only in the neighbouring cells instead of considering all particles
- Dirty flag ?

POTENTIAL MODULE
description how the particles will interact in the simulation; two main approximations: Born–Oppenheimer approximation (dynamics of electrons), point particles that follow classical Newtonian dynamics (nuclei, which are much heavier than electrons)

Coulomb-Potential (-> Particle-Mesh)

$$\phi_i(\mathbf{r}) = \frac{1}{4\pi\varepsilon_0}\frac{q_i}{|\mathbf{r}-\mathbf{r_i}|}$$

$$\frac{1}{4\pi\varepsilon_0} = c_0^2\mu_0 = {}^{c^2_0\mu_0} = c^2{}_0 \times 10^{-7}\,\mathrm{H\cdot m^{-1}} = 8.987\times 10^9\,\mathrm{Nm^2C^{-2}}$$

Lennard-Jones

Particle-mesh Ewald

Ewald summation

Usage: standard method for calculating long-range interactions in periodic systems
- rapid convergence of the energy compared with that of a direct summation: high accuracy and reasonable speed when computing long-range interactions
-

INTEGRATOR MODULE
See notebooks :
- Euler
- Langevin
- Velocity-verlet

MINIMISATION

???

Lecture notes

line_profiler
Piecewise
@njit
Recent body: not can chance its structure , shake or radle
Hard spheres
Ewald analyticaly fouler (strait forward)
Particle mesh a numerical fouler (corektli sparred particle)
Gommecs

codacy : https://app.codacy.com/login
Compile?
Circle ci

Präsentation