
6.VAJA: MONTE CARLO SIMULACIJA

Dominik Primožič

23. APRIL 2025

1. naloga

Napišite program, ki vam naredi Monte Carlo simulacijo za 1D, 2D in 3D Lennard-Jonesove delce. Poglejte, kako se energija in porazdelitvena funkcija spreminja kot funkcija temperature in gostote. Poglejte si tudi snapshote (slike) sistema ob različnih trenutkih.

Metode

Monte Carlo simulacije sem se lotil v več sklopih- iniciacija sistema, naključni premiki sistema, lokalni izračuni spremembe energije. Za iniciacijo sem uporabil, kar objekt iz prvih vaj, ki naključno porazdeli delce v škatli, pri čemer sem kot premer delcev vzel kar σ . Za evolucijo sistema sem iz vektorja koordinat izbral naključen indeks in za naključen korak spremenil koordinate pripadajočemu delcu. Pred in po spremembi sem izračunal interakcije tega delca s preostalimi delci v sistemu z Lennard-Jonesovim parskim potencialom.

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right]$$

To spremembo energije sem nato vnesel v Metropolisov algoritem, kjer se potrdi ali je premik sprejet ali ne. Med ekvilibracijo sistema se na vsakih nekaj premikov preveri delež sprejetih premikov, glede na to se prilagaja velikost koraka, da je sprejetih premikov med 40 in 60%. Med sampling potekom se koraka ne spreminja več, se pa na vsakih nekaj izbranih korakov zapisuje energija in konfiguracija sistema. Iz konfiguracij sem na koncu izračunal povprečne porazdelitvene funkcije. Program deluje identično za vse tri dimenzije, kot vhod pa sprejme datoteko s parametri.

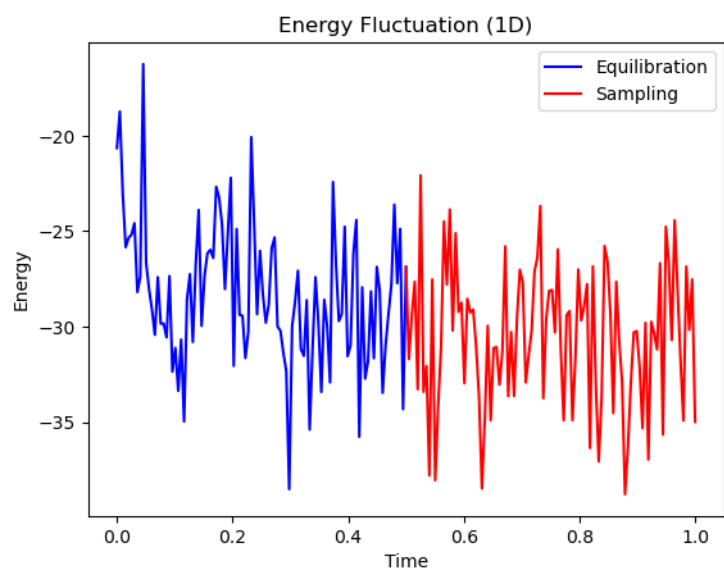
V izračunih sem uporabil reducirane enote ($\sigma=1$, $\epsilon=1$, $T = \frac{T k_b}{[\epsilon]}$)

Rezultati

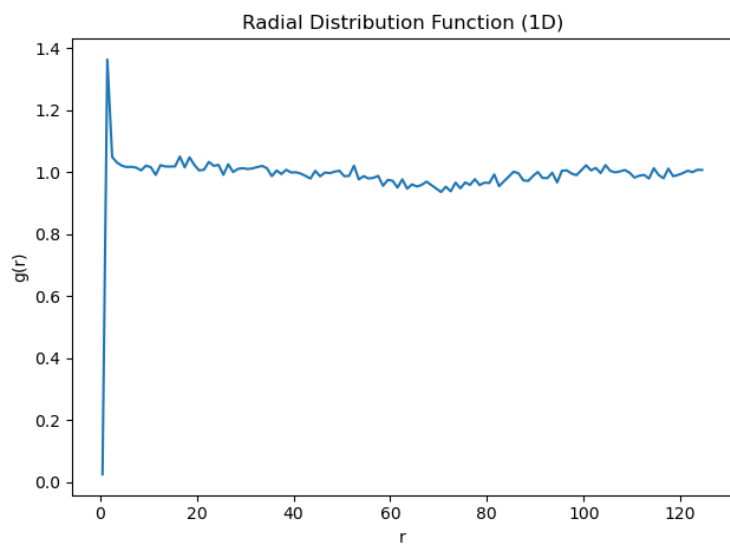
Za osnove parametre vseh simulacij sem izbral naslednje vrednosti: N (število delcev) = 100, ρ (gostota) = 0.4, dimenzije = 2, temperatura = 1, ϵ (energijski parameter) = 1, σ (dolžinski parameter) = 1, cutoff radij = 2.5σ .

Za izvedbo simulacije sem uporabil naslednje časovne nastavitve: koraki ekvilibracije = 1000000, koraki vzorčenja = 1000000, koraki uglaševanja = 1000, preverjanje stanja ekvilibracije vsakih 1000 korakov ter zajem podatkov (sampling) vsakih 1000 korakov.

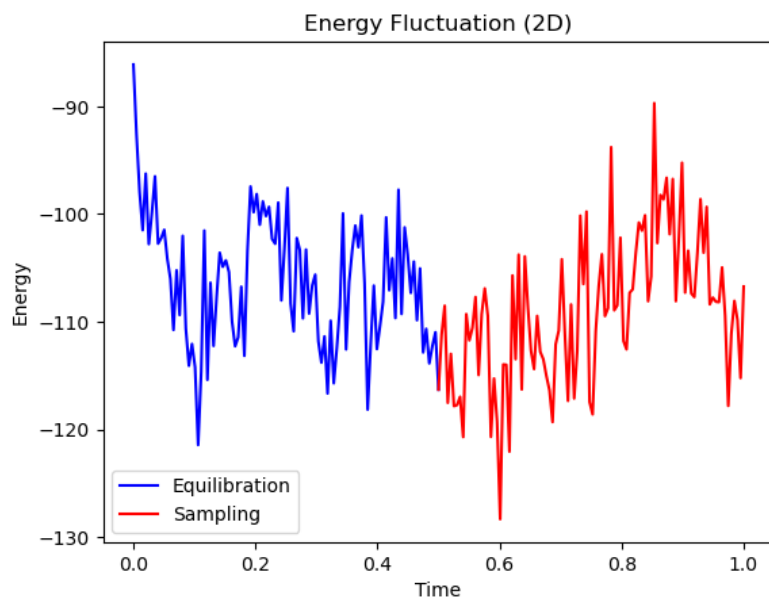
Sprva sem za demonstracijo simulacijo pognal na teh parametrih za vse tri dimenzije. Spodaj je povprečna energija skozi celoten čas simulacije in povprečna radialna funkcija na zajetih konformacijah.



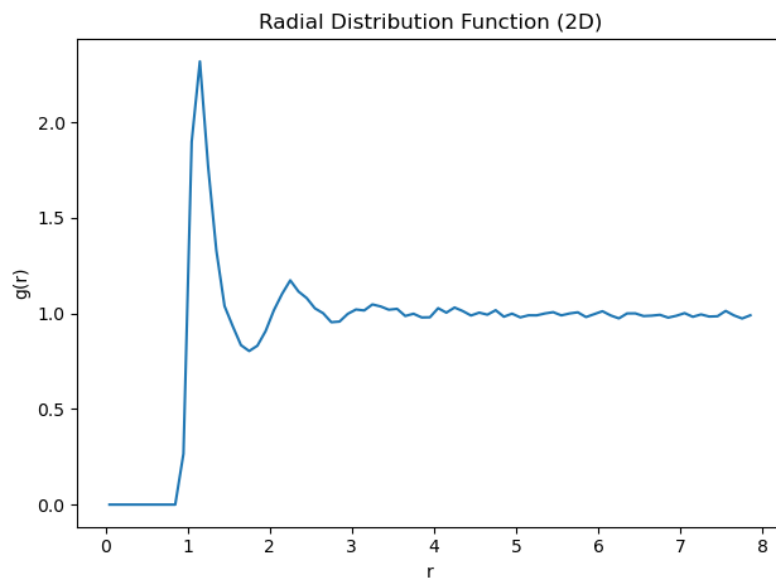
Graf 1: Povprečna energija za 1D LJ delce



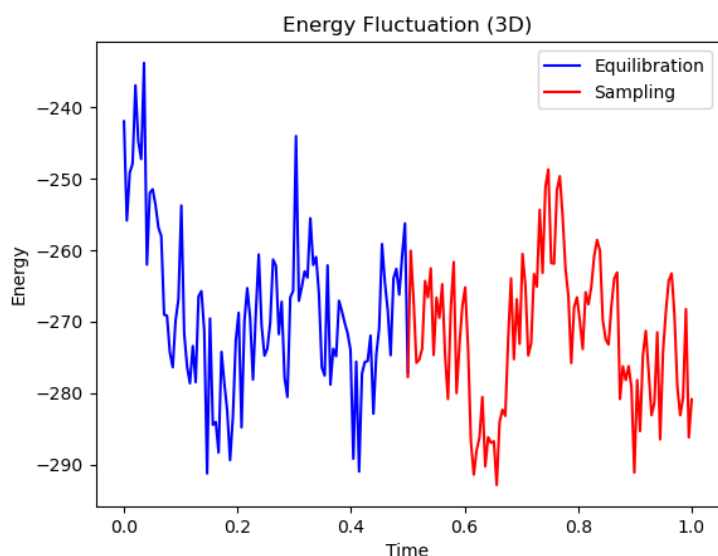
Graf 2: Povprečna porazdelitvena funkcija za 1D LJ delce



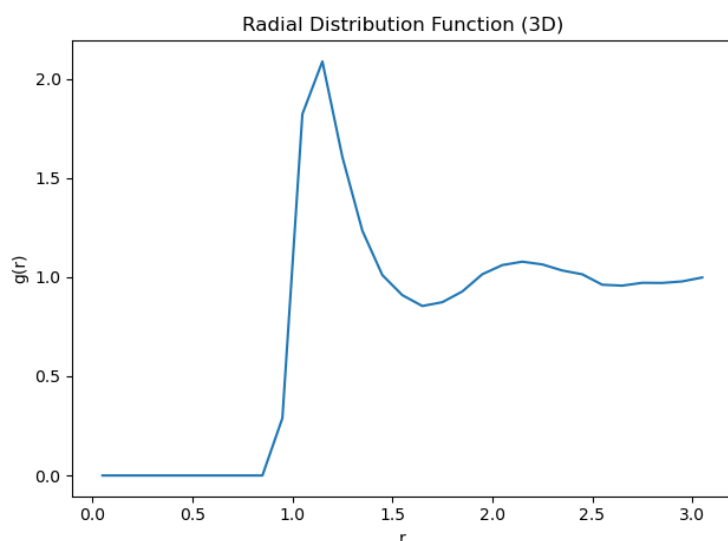
Graf 3: Povprečna energija za 2D LJ delce



Graf 4: Povprečna porazdelitvena funkcija za 2D LJ delce



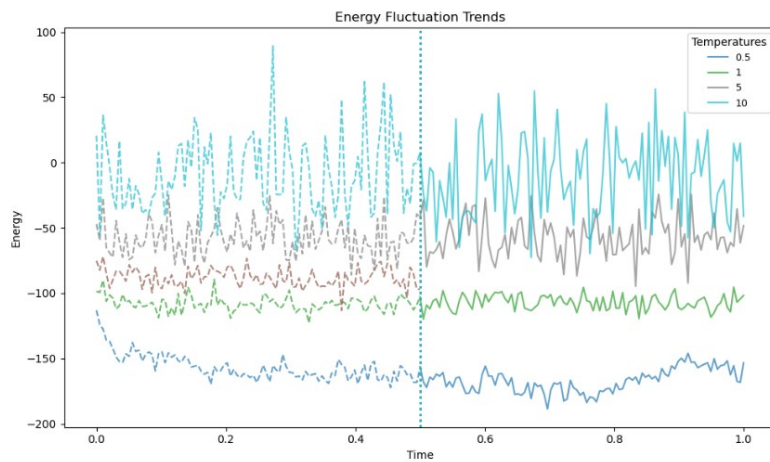
Graf 5: Povprečna energija za 3D LJ delce



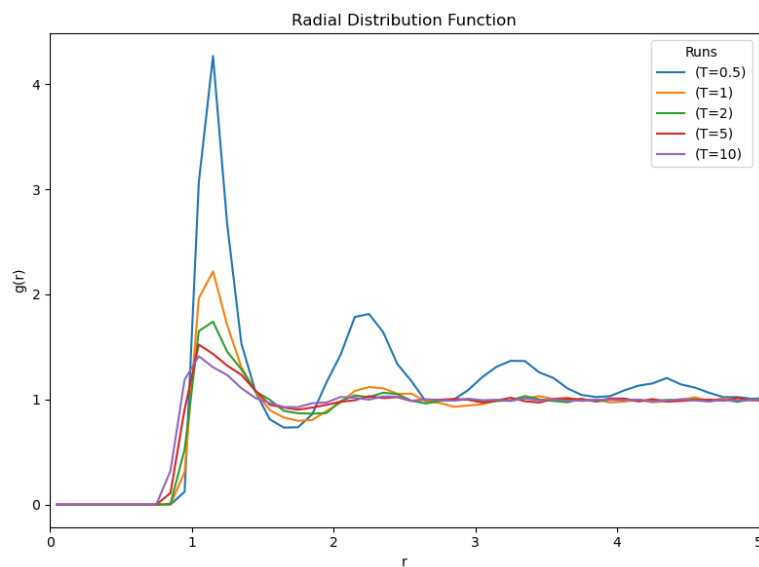
Graf 6: Povprečna porazdelitvena funkcija za 3D LJ delce

Energija se nekako ustali pri vseh, čeprav ne izgleda. Tudi v primeru, ko sem povečal število ekvilibracijskih korakov za faktor 100 je bilo nihanje približno enako. Nihanje je sicer le za približno 10% povprečne energije, kar ni slabo. Porazdelitvena funkcija najbolje kaže vpliv privlačnih interakcij, z dvema lupinama delcev, v treh dimenzijah, najslabše pa v eni dimenziji. Za več delcev rezultatov porazdelitvene funkcije nisem prikazal grafično, ker je obdelava podatkov trajala predolgo.

Nato sem preveril vpliv temperature na povprečno energijo in porazdelitveno funkcijo pri sistemu 2D delcev.



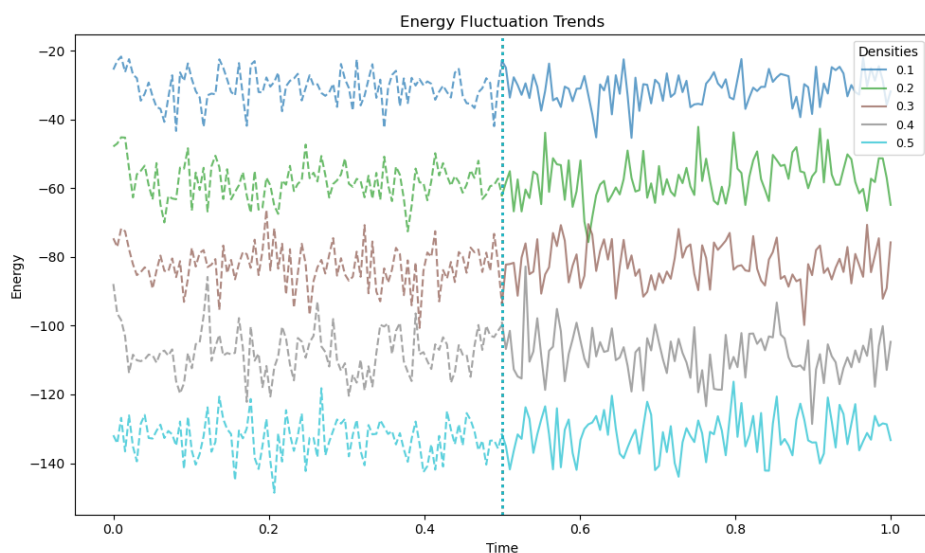
Graf 7: Vpliv temperature na povprečno energijo



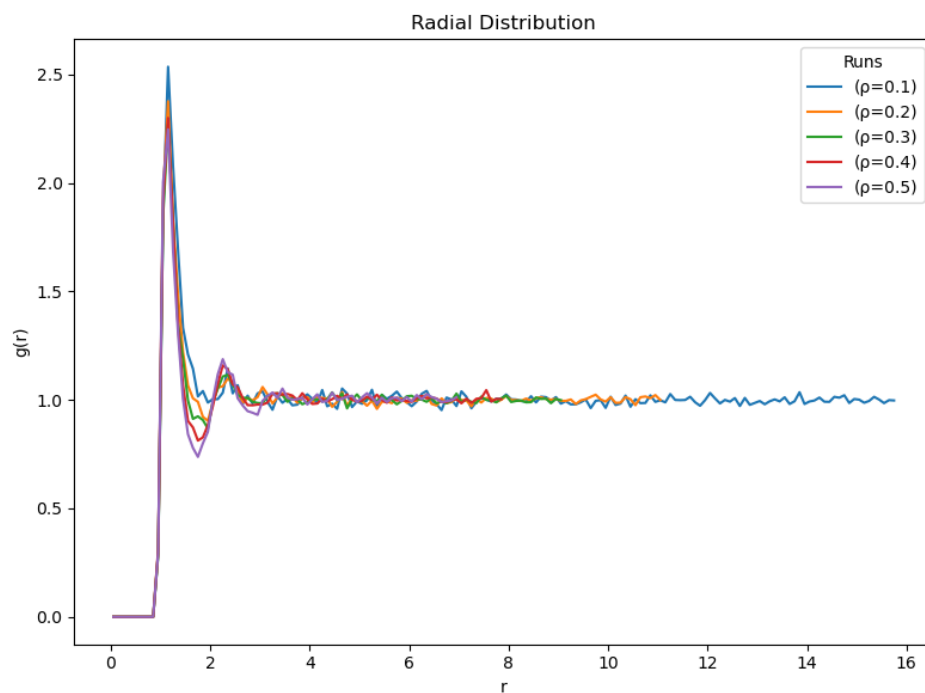
Graf 8: Vpliv temperature na porazdelitveno funkcijo

Kot pričakovano je povprečna temperatura višja pri višji temperaturi in tudi njene fluktuacije so večje. To je v skladu s pričakovanji gibanja delcev z visoko termično energijo. Podobno je opazno tudi v porazdelitveni funkciji, kjer z višjo temperaturo izgublamo urejenost v lupinah. Pri nizki temperaturi so delci lepo urejeni v 4 lupine, pri visoki temperaturi pa je le majhna privlačna urejenost na prvi lupini, torej delci imajo dovolj termične energije, da privlaka ne čutijo več na razdalji izven minimuma LJ funkcije.

Nato sem preveril še vpliv spreminjanja gostote.



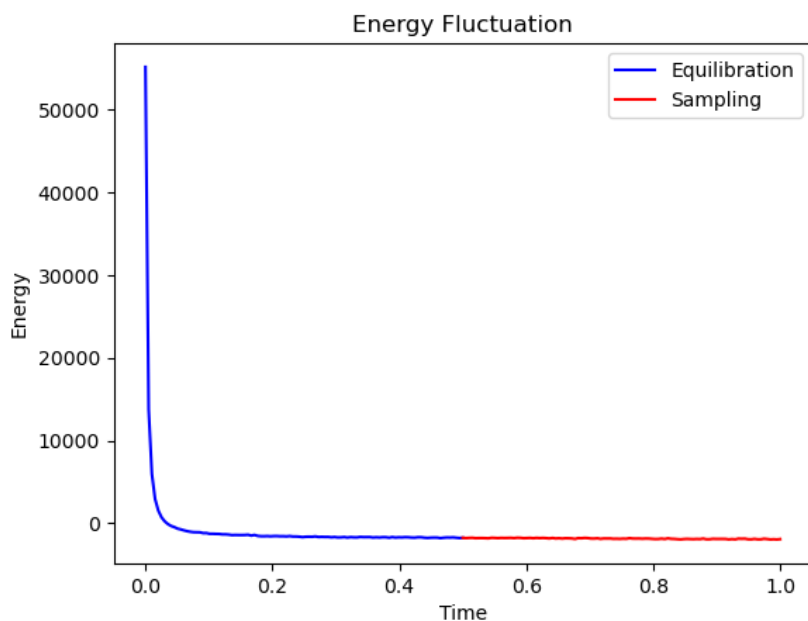
Graf 9: Vpliv gostote na povprečno energijo



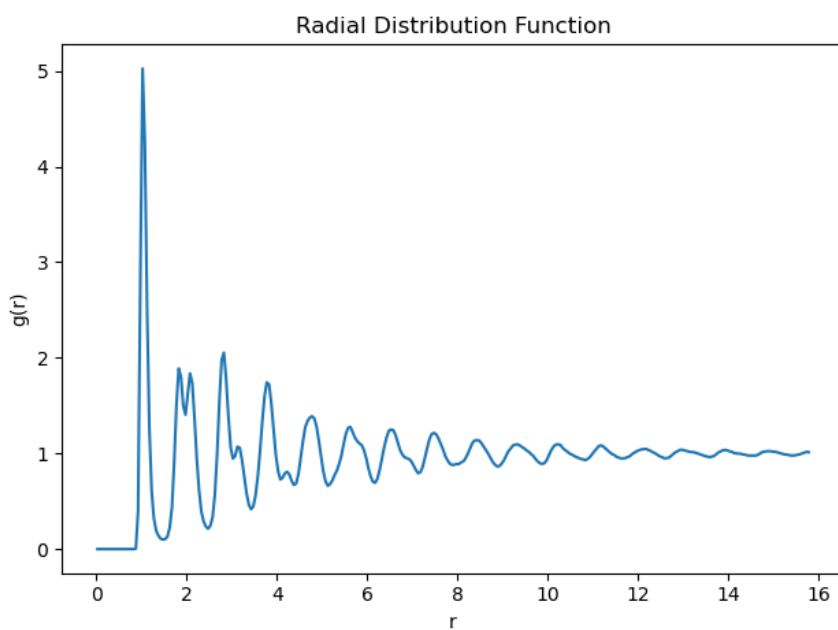
Graf 10: Vpliv gostote na porazdelitveno funkcijo

Pri večji gostoti so delci bolj skupaj blizu in čutijo več privlačnih interakcij, zato imajo nižjo energijo, kar je lepo razvidno iz grafa 9. Podobno a manj izrazito se pri porazdelitveni funkciji vidi večja urejenost delcev (več lupin) pri večji gostoti.

Želel sem simulirati tudi tekoče stanje, to sem storil tako, da sem delce postavil bolj na gosto. Dovolil sem, da so na razdalji najmanj 0.5σ z gostoto 1.



Graf 11: LJ tekočina, povprečna energija



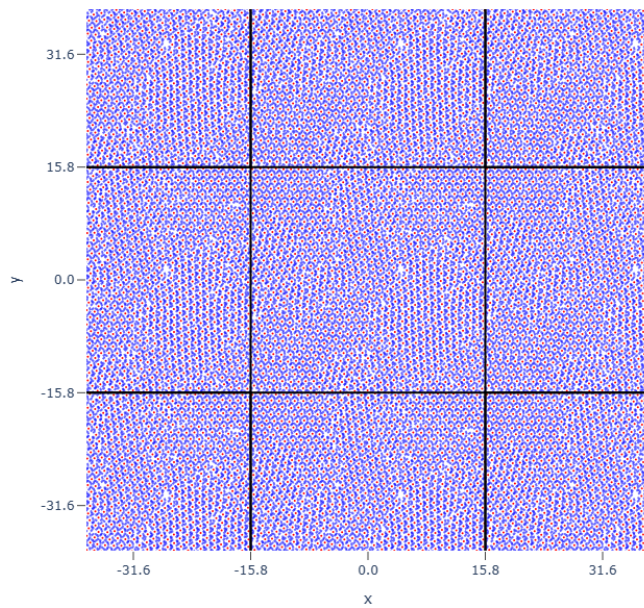
Graf 12: LJ tekočina, porazdelitvena funkcija

Povprečna energija je na začetku zelo velika, a se počasi ustali v rahlo negativni vrednosti (-1.8 na delec). Pri porazdelitveni funkciji se pojavi več lupin, kar kaže na večjo urejenost, ki je značilna za tekočine.

Na koncu pa sem naredil še nekaj slik 2D sistema ob različnih trenutkih. Z modro obrobo sem označil navidezni radij σ , torej radij odbojne interakcije, z rdečo pa središče delca.

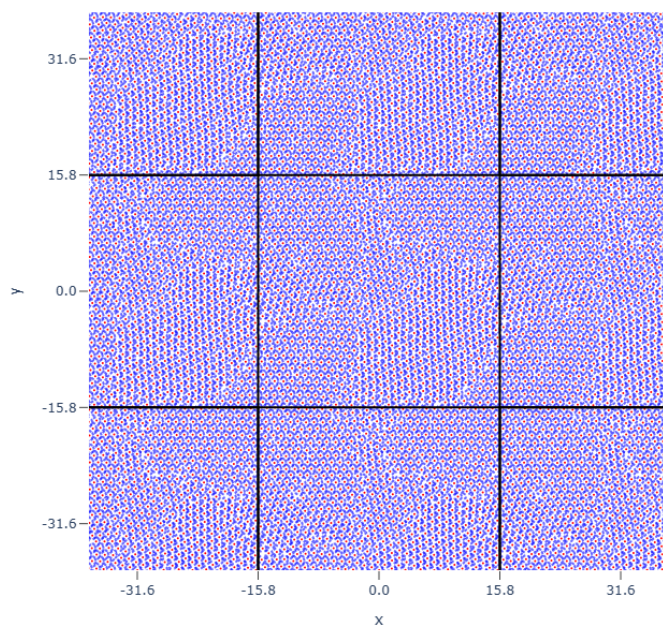
Najprej sem naredil sliko prejšnjega tekočega sistema na začetku in na koncu.

Snapshot at step_0



Graf 13: Slika LJ tekočine na začetku

Snapshot at step_990000

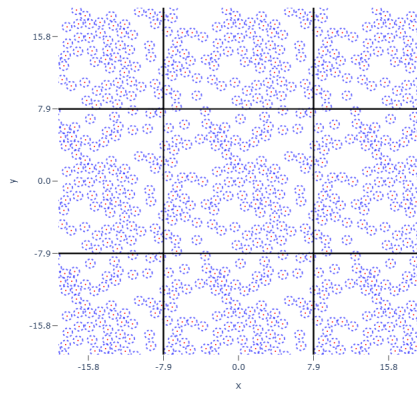


Graf 14: Slika LJ tekočine na koncu

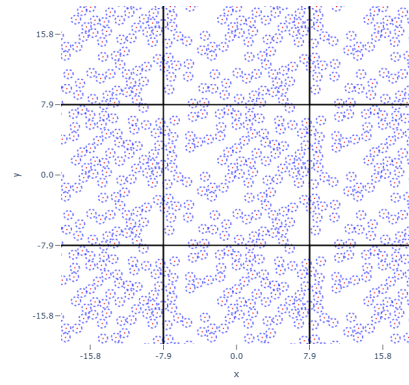
Jasno je razvidno, da gre za zelo gosto postavljene delce. Na zadnji sliki sistema se tudi vidi, da se je sistem uredil, ni več lukenj, ki so bile prisotne na začetku zaradi naključne porazdelitve.

Naslednje slike so za sistem pri osnovnih parametrih ob različnih korakih simulacije.

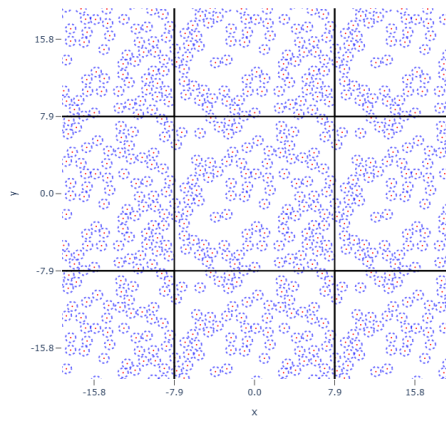
Snapshot at step_0



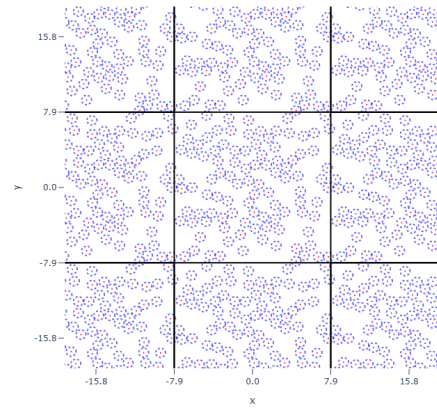
Snapshot at step_200000



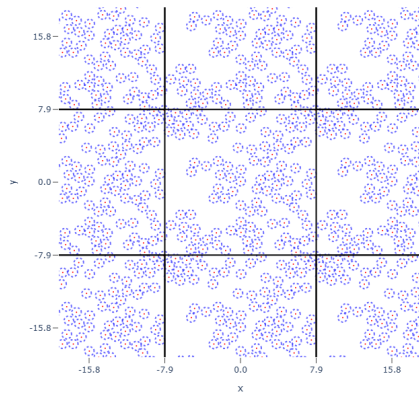
Snapshot at step_400000



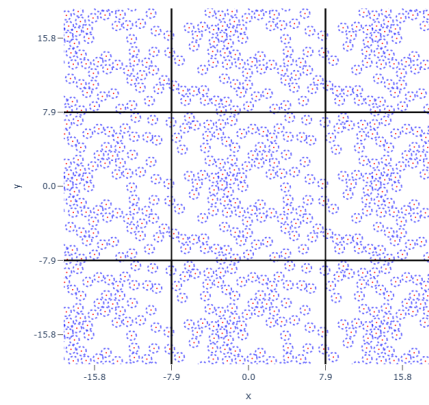
Snapshot at step_600000



Snapshot at step_800000



Snapshot at step_900000



Graf 15: Slike sistema ob različnih simulacijskih korakih

2. naloga

Uporabite Metropolisov algoritem Monte Carlo metode za vzorčenje gostote elektronov v preprostem 1D paraboličnem potencialu. Izračunajte pričakovano vrednost položaja in energije elektrona.

Metode

Naloge sem se lotil z variacijsko Monte Carlo metodo. Za valovno funkcijo sem vzel Gaussovo funkcijo:

$$\psi(r) = \left(\frac{a}{\pi}\right)^{1/4} e^{-\frac{1}{2}ar^2}$$

za katero lahko analitično izračunamo lokalno energijo:

$$E_{\text{loc}}(r) = \frac{a}{2} + \left(\frac{1}{2} - \frac{a^2}{2}\right)r^2$$

Energijo sistema se po MC metodi izračuna z naključnim spreminjanjem pozicije za korak δ , tako da se uspešnost preverja glede na Metropolisov algoritem:

$$r' = r + 2\delta(\xi - 0.5), \quad \xi \sim \mathcal{U}(0,1)$$

$$P_a = \min\left(1, \frac{|\psi(r')|^2}{|\psi(r)|^2}\right)$$

Dolžino koraka se prilagaja, da je delež sprejetih korakov med 40% in 60%. Povprečne vrednosti se izračuna enako kot v običajnem MC, na vsakih nekaj korakov se shrani vrednost energije, ki jo potem povprečimo na vse korake.

Optimalno vrednost parametra a sem izračunal z zlatim rezom.

Samo simulacijo sem napisal v paralelni obliki

```
double electron_in_potential::run(double a, int steps, double delta, int
discard, int sample_interval, int tune){
    double energy = 0.0;
    expected_r = 0.0;
    expected_r2 = 0.0;
    int sampled = 0;
    #pragma omp parallel
    {
        std::mt19937 thread_gen(gen());
        double local_r = 0;
        double local_energy_sum = 0;
        double local_r_sum = 0;
        double local_r2_sum = 0;
        int local_sampled = 0;
        int taccept = 0, ttry = 0;
```

```

for (int i = 0; i < discard; ++i) {
    if (metropolis(local_r, a, delta)) taccept++;
    ttry++;
    if (ttry==tune){
        double A = double(taccept) / ttry;
        if (A > 0.4) {
            delta *=1.05;
        } else if (A < 0.6){
            delta *=0.95;
        }
        taccept = ttry=0;
    }
}
for (int i = 0; i < steps; ++i) {
    metropolis(local_r, a, delta);
    if (i % sample_interval == 0) {
        double e = local_energy(local_r, a);
        local_energy_sum += e;
        local_r_sum += local_r;
        local_r2_sum += local_r * local_r;
        ++local_sampled;
    }
}
#pragma omp critical
{
    energy += local_energy_sum;
    expected_r += local_r_sum;
    expected_r2 += local_r2_sum;
    sampled += local_sampled;
}
expected_r /= sampled;
expected_r2 /= sampled;
return energy / sampled;
}

```

Rezultati

Za različno število korakov simulacije sem izračunal povprečne vrednosti. Vzorčenje sem izvajal na najmanj 1/1000 korakov simulacije.

Število korakov	α	E	$\langle x \rangle$	var(x)
1000	1.0139	0.500611	-0.00512077	0.6729407683
10000	0.971936	0.500595	-0.0127749	0.7269400263
100000	1.01025	0.500063	-0.00461132	0.7009156409
1000000	1.01193	0.500037	-0.0153616	0.7026756159
10000000	1.00742	0.500005	-0.0153616	0.7051709163

Tabela 1: Rezultati variacijskega MC

V vseh primerih je alfa dokaj blizu 1, kar je tudi analitična rešitev. Energija je v vseh primerih praktično 0.5, kar je tudi s skladu z analitično rešitvijo. Pričakovana vrednost elektrona bi morala biti 0, v vseh izračunih pa je majhen negativni odmik, a šele na drugi decimalki, kar ni tako slabo. Teoretična vrednost $\langle x^2 \rangle$ je 0.5, kar naredi teoretično vrednost variance približno 0.707. V simulaciji z večjim številom korakov je rezultat zelo blizu tej vrednosti, pri manj korakih pa je odstopanje veliko, ker gre za korelacijo med vzorčenimi stanji.