5. Molekuladinamika

Csillag Barnabás Gellért (COTNU3)

2020. április 20.

1. Az elméleti háttér, és a megoldási módszerek

Jelen jegyzőkönyv célja a nagy számú mikroszkopikus részecskék viselkedésének szimulálásával makroszkopikus rendszerek leírását célzó algoritmusok vizsgálata.

A mai számítógépekkel nem lehetséges makroszkopikus részecskeszámú rendszerek direkt leírása. Azonban megfelelő elhanyagolásokkal, feltevésekkel sok részecskéből álló mikroszkopikus rendszerek viselkedésével közelíthetünk makroszkopikus rendszereket, és ilyen módon meghatározhatjuk a termodinamikai tulajdonságaikat. Ha a részecskék közt ható erő gyorsan lecseng, az $\frac{N(N-1)}{2} \sim N^2$ darab párkölcsönhatásra az alábbi közelítéseket vezethetjük be:

- a távoli erők nagy térrészekre kiátlagolódnak,
- a nagyon távol eső részecskék kölcsönhatása elhanyagolható.

Kezdetben van N darab részecske, mindegyiknek ismerjük a helyét és a sebességét - tipikusan nemegyensúlyi helyzetből indulunk. Kiszámoljuk a részecskék között ható erőt, és ez alapján léptetjük a rendszert. Idővel a rendszernek a tranziens szakaszból egy egyensúlyi állapotba kell konvergálnia. Egyensúlyi állapotban teljesül az ekvipartíció tétele, vagyis egyatomos molekuláknál szobahőmérsékleten minden részecskére $< K >= \frac{3}{2} k_B T$ kinetikus energia jut - egy szimuláció során ezzel mérhető, hogy egyensúlyban van-e a rendszer. Az olyan mennyiségeket, mint hőkapacitás, nyomás vagy hőmérséklet, egyensúlyban lehet megmérni.

Precíz számításokhoz az atomok belső szerkezetét figyelembe vevő kvantummechanikát kellene alkalmazni, de ha valamilyen nemesgázt, például argont vizsgálunk, akkor a klasszikus közelítés is elfogadható pontosságú eredményekre vezet. Ennek oka, hogy szobahőmérsékleten az argon atomok kinetikus energiájánál háromszázszor nagyobb a gerjesztéshez szükséges energia, és a de Broglie hullámhosszuk is jelentősen kisebb, mint a részecskék effektív mérete vagy egymás közti távolsága.

A termodinamikai egyensúly eléréséhez a szimuláció lépéshosszát kisebbnek kell választani, mint amennyi idő alatt a részecskék meg tudnák tenni az egymás közti távolságot. Ez argon atomok esetében kevesebb, mint egy pikoszekundum, vagyis ezer atom néhány nanoszekundum ideig történő szimulációja is órákig tarthat - ezért fontos a megfelelő kezdőfeltételek beállítása.

A legelterjedtebb molekuladinamikai szimulációs módszerek a Verlet-, és a velocity-Verlet-algoritmus. A **Verlet-algoritmus** léptetési szabályai:

$$\overrightarrow{R}_{n+1} = 2\overrightarrow{R}_n - \overrightarrow{R}_{n-1} + \tau^2 \overrightarrow{A}_n + \mathcal{O}(\tau^4), \tag{1}$$

$$\overrightarrow{V}_n = \frac{\overrightarrow{R}_{n-1} + \overrightarrow{R}_{n+1}}{2\tau} + \mathcal{O}(\tau^2), \tag{2}$$

ahol az n-edik szimulációs lépésben $\overrightarrow{R}_n = (\underline{r}_1, ..., \underline{r}_N)_n$ a részecskék koordinátáit, $\overrightarrow{V}_n$ a részecskék sebességeit, $\overrightarrow{A}_n$ pedig a részecskék gyorsulásait jelöli N részecske esetén.

A módszer **előnyei**:

- Gyorsabb, mint a Runge-Kutta metódusok, mert egy lépés során csak egyszer kell kiszámolni a gyorsulásokat.
- Majdnem olyan pontos, mint a Runge-Kutta, ahol $\mathcal{O}(\tau^5)$ hiba van.
- Jól megőrzi az energiát.
- Időtükrözésre szimmetrikus, ami fontos a részletes egyensúly elve miatt.

A módszernek három jelentős **hátránya** van:

- Két előző lépést használ, vagyis nem indítható egy kezdeti feltétellel.
- A sebesség hibája $\mathcal{O}(\tau^2)$, de ez csak sebességtől függő erők esetén rontja számottevően a pozíciók pontosságát.
- A sebesség és a hely nem egyszerre frissül, így a sebesség lemarad.

A **velocity-Verlet-algoritmus** léptetési szabályai:

$$\overrightarrow{R}_{n+1} = \overrightarrow{R}_n + \tau \overrightarrow{V}_n + \frac{\tau^2}{2} \overrightarrow{A}_n + \mathcal{O}(\tau^3), \tag{3}$$

$$\overrightarrow{V}_{n+1} = \overrightarrow{V}_n + \frac{\tau}{2}(\overrightarrow{A}_{n+1} + \overrightarrow{A}_n) + \mathcal{O}(\tau^3). \tag{4}$$

A módszer előnyei a Verlet-algoritmushoz képest, hogy nem követel meg időben két korábbi lépést, egyszerre frissíti a sebességeket és a koordinátákat, továbbá a sebesség pontosabb. Hátránya, hogy a pozíció itt csak $\mathcal{O}(\tau^3)$-ben pontos.

A részecskék közti kölcsönhatást a van der Waals-közelítésben a **Lennard-Jones potenciál** írja le:

$$V(r) = 4V_0 \left[\left(\frac{r_0}{r} \right)^{12} - \left(\frac{r_0}{r} \right)^{6} \right], \tag{5}$$

ahol r a két részecske közti távolságot jelöli, r_0 és V_0 pedig a részecske típusától függő konstansok. A képlet alapján is leszűrhető, hogy a Lennard-Jones potenciál $r = r_0$-nál előjelet vált, $r = 2^{1/6} r_0$-nál veszi fel a minimumát, ami $-V_0$, erősen taszító kis távolságokon (a Paul-elv miatt), és enyhén vonzó nagy távolságokon.

Az erő legegyszerűbben a potenciál gradienséből számolható:

$$\overrightarrow{R}(\underline{r}) = -\underline{\nabla} V(r) = \frac{24 V_0}{r^2} \left[2 \cdot \left(\frac{r_0}{r} \right)^{12} - \left(\frac{r_0}{r} \right)^{6} \right] \underline{r}. \tag{6}$$

A rendszer energiája pedig

$$E = \sum_{i=1}^{N} \frac{1}{2} m_i v_i^2 + \sum_{<ij>} V(r_{ij}). \tag{7}$$

Egyensúlyban a sebességek Maxwell-Boltzmann eloszlást követnek.

Hogyha valós paramétereket használnánk, a rendszer karakterisztikus idejére $\tau_0 = \sqrt{\frac{m r_0^2}{V_0}} = 2.17 \cdot 10^{-12} s$ jönne ki. Innen látszik, hogy nem érdemes SI értékekkel szimulációt futtatni, mert numerikusan instabil lenne. Ehelyett megválaszthatjuk a skálát úgy, hogy $m = 1$, $r_0 = 1$, $V_0 = 1$ legyen, így $\tau = 1$.

Konkrét három dimenziós Lennard-Jones rendszerek szimulációja esetén érdemes **periodikus határfeltételek**et választani, hogy ne kelljen speciális feltételekkel (pl. fallal ütközés) foglalkozni, ugyanakkor állandó maradjon a részecskeszám. Így tehát a periodikus határfeltétel esetén ami részecske az egyik oldalon kilép a valahány dimenziós dobozból, az a másik oldalon ugyanott belép.

Ha a tranziens szakasz viselkedését szeretnénk vizsgálni, akkor indíthatjuk a részecskéket egyszerű köbös rácsból, a sebességeket egyenletes eloszlás alapján generálva. Ha viszont a lehető leggyorsabban el akarjuk érni az egyensúlyt, akkor indíthatjuk a részecskéket lapcentrált köbös rácsból, Maxwell-Boltzmann-sebebességeloszlással:

$$P(|\underline{v}|) = \left(\frac{m}{2\pi k_B T}\right)^{3/2} e^{-\frac{m(v_x^2+v_y^2+v_z^2)}{2k_B T}}. \tag{8}$$

Az md2.cpp, md3.cpp kódokban a sebességek véletlen generálása a *Numerical Recipes*-ből átvett *gasdev*() függvény által történt, amely nulla várható értékű, egy szórású eloszlással rendelkező számokat ad vissza, és a Box-Müller-algoritmuson alapszik. Mivel az így generált kezdősebességek átlaga nem egzaktul nulla, a tömegközépponti sebesség levonásával korrigálni kell a sebességeket: $\underline{v}_i \to \underline{v}_i - \underline{v}_{tkp}$, ahol a tömegközépponti sebesség

$$\underline{v}_{tkp} = \frac{\sum_{i=1}^{N} m\underline{v}_i}{\sum_{i=1}^{N} m}. \tag{9}$$

Ezt követően az egy szórású sebességeket a kívánt hőmérsékletre kell átskálázni úgy, hogy a T hőmérsékletet kapjuk: $\underline{v}_i \to \lambda \underline{v}_i$, ahol

$$\lambda = \sqrt{\frac{2(N-1)k_B T}{\sum_{i=1}^{N} m\underline{v}_i^2}}. \tag{10}$$

Az erők páronkénti kölcsönhatása nagyon időigényes ($t \sim N^2$). A helyzetet javíthatja, ha különböző gyorsítást célzó változtatásokat teszünk az algoritmusban. Mivel a Lennard-Jones erő rövid hatótávú, bevezethető egy úgynevezett r_{cutoff} távolság, amelyen túl az erő hatása praktikusan nullának tekintendő. Így, ha a sűrűség állandó, akkor a páronkénti kölcsönhatások számítása nem négyzetesen fog skálázni a részecskék számával, hanem lineárisan. A módszer akkor tud hasznos lenni, ha $r_{cutoff} << L$, ahol L a rendszer mérete.

Azonban a levágás érvényesítéséhez mindig tudni kellene azt is, hogy melyek azok a részecskék, amelyekre $r_{ij} > r_{cutoff}$. Mivel minden részecske mozog, ezen művelet számításigénye is N^2-el skáláz. Ha viszont figyelembe vesszük, hogy lépésenként a részecskék csak keveset mozdulnak el, akkor lehet választani egy $L >> r_{max} > r_{cutoff}$ távolságot, amelynél jobban néhány lépésen belül nem távolodnak el a részecskék, így elég azon belül nyilvántartani őket, és néhány lépésenként frissíteni a nyilvántartást. Ezen paraméterekre a legideálisabb választás az $r_{cutoff} = 2.5r_0$, $r_{max} = 3.2r_0$, és a tíz lépésenkénti frissítés lehet.

Egy ilyen molekuladinamikai szimuláció során a következő **termodinamikai mennyiségek** határozhatók meg:

- A hőmérséklet. Ez egyensúlyban a kinetikus energiából származtatható az ekvipartíció tételének segítségével, amely a következőképpen néz ki nulla átlagsebesség esetén:

$$3(N-1)\cdot\frac{1}{2}k_BT=\langle\frac{m}{2}\sum_{i=1}^{N}v_i^2\rangle, \tag{11}$$

  ahol $\langle ...\rangle$ a termikus sokaság átlagot jelöli, $3(N-1)$ pedig a belső szabadsági fokok száma. Nemegyensúlyi helyzetben nem teljesül az ekvipartíció tétele, de lemérhetjük általa T mennyiséget, mint pillanatnyi hőmérsékletet.

- A teljes energia:

$$E=\sum_{i=1}^{N}\frac{1}{2}mv_i^2+\sum_{i\neq j}V(r_{ij}). \tag{12}$$

- A hőkapacitás a fluktuáció-disszipáció tétel alapján:

$$C_V=\frac{1}{k_BT^2}(\langle E^2\rangle-\langle E\rangle^2). \tag{13}$$

  Az átlagolás általános esetben sokaságátlagot jelent, vagyis több különböző futtatás eredményét kellene felhasználni hozzá, de egyensúlyban ezt jól közelíti az időátlag.

- A nyomás kiszámításához felhasználhatjuk a Viriál-tételt:

$$PV=Nk_BT+\frac{1}{3}\langle\sum_{i<j}\underline{r}_{ij}\cdot\underline{F}_{ij}\rangle. \tag{14}$$

- A kompresszibilitási faktor, amely kifejezi, hogy a rendszer tulajdonságait tekintve mennyire van távol az ideális gáztól:

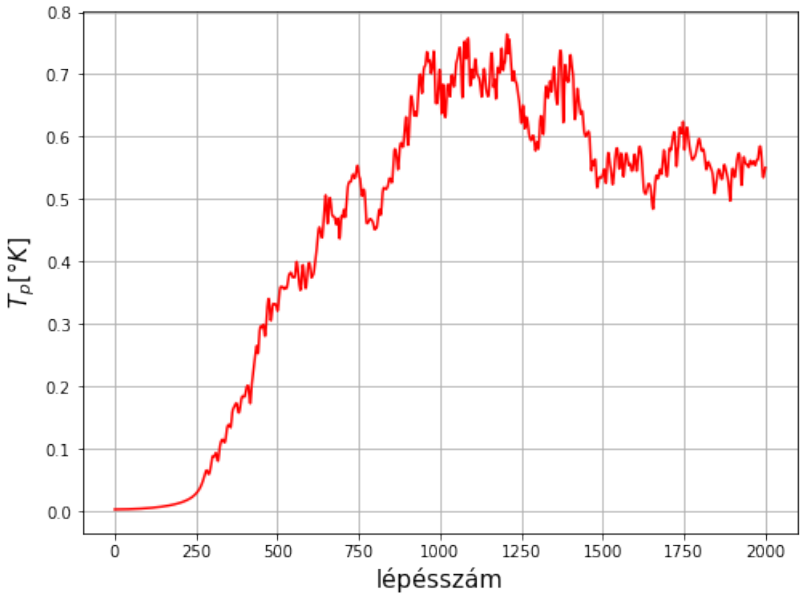
$$Z=\frac{PV}{Nk_BT}=1+\frac{1}{3Nk_BT}\langle\sum_{i<j}\underline{r}_{ij}\cdot\underline{F}_{ij}\rangle. \tag{15}$$

  Nagy sűrűségre a taszító potenciál miatt $Z>1$, kis sűrűségen pedig a vonzó van der Waals-erők miatt $Z<1$, míg ideális gáz esetében $Z=1$.

2. Eredmények

2.1. Az md.cpp és az md2.cpp összehasonlítása

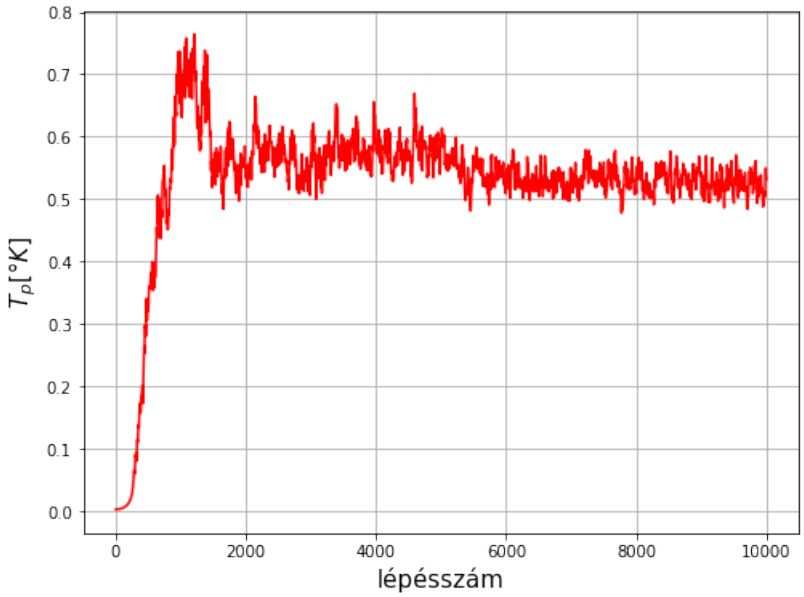
Az első program esetében a részecskék kezdeti pozíciói egy egyszerű köbös rácsot formáznak. A kezdeti sebességek egyenletes eloszlás alapján generálódnak. A szimuláció tehát a velocity-Verlet-algoritmust valósítja meg, de nem tartalmaz gyorsításokat, és periodikus határfeltételeket sem. Az előre megírt szimuláció minden lépés során kiszámítja a pillanatnyi hőmérsékletet a (11)-es egyenlet alapján.



1. ábra. Az md.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, és $n_{max} = 2000$ maximális lépésszám mellett úgy, hogy a teljes rendszer kezdő mérete egy $L = 10m$, és $N = 64$ részecskét tartalmaz.

Az időnek, a hossznak és a hőmérsékletnek az átskálázás miatt valamilyen nehezen értelmezhető mértékegysége kellene, hogy legyen - én most azért, hogy ne maradjanak dimenzió nélkül, szekundumot, métert és Kelvint fogok írni, viszont észben kell tartani, hogy ezek most nem SI mértékegységek.

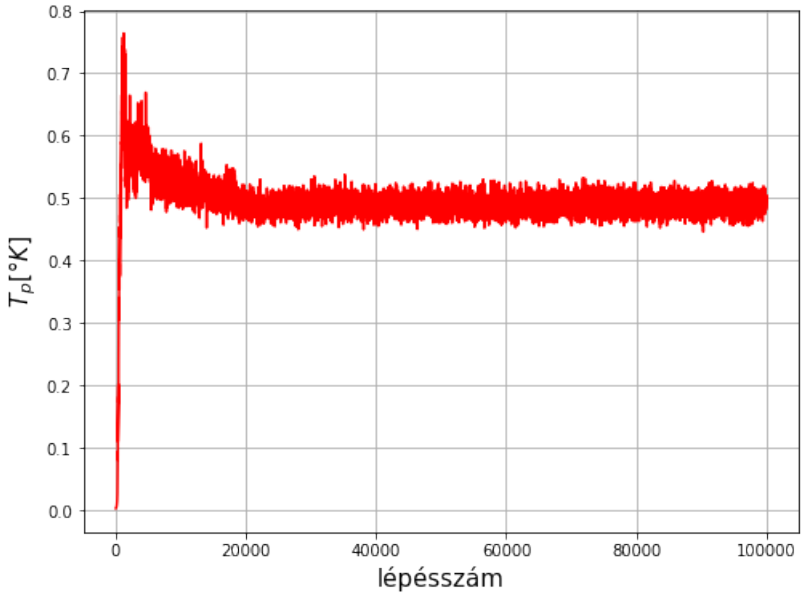
Látható, hogy a pillanatnyi hőmérséklet a kezdeti gyors emelkedés után $0.5 - 0.6K$ körül ingadozik. A tranziens szakasz szemre körülbelül ezer lépésig tart - futtassunk hát egy szimulációt tízezerig, majd átlagoljuk ezertől tízezerig a pillanatnyi hőmérsékletet - így elméletileg meg kellene kapnunk az egyensúlyi hőmérsékletet. Az adatok szórásából kiszámíthatjuk a hőmérséklet hibáját is.



2. ábra. Az md.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, és $n_{max} = 10000$ maximális lépésszám mellett úgy, hogy a teljes rendszer kezdő mérete egy $L = 10m$, és $N = 64$ részecskét tartalmaz.

Az látható, mint az előbb - a kezdeti gyors emelkedés után a hőmérséklet $0.5 - 0.6K$ körül ingadozik. A kiszámított egyensúlyi hőmérséklet hibával: $T_{egy} = 0.5553K \pm 0.0017K$.

Mehetünk tovább is a szimulációval:

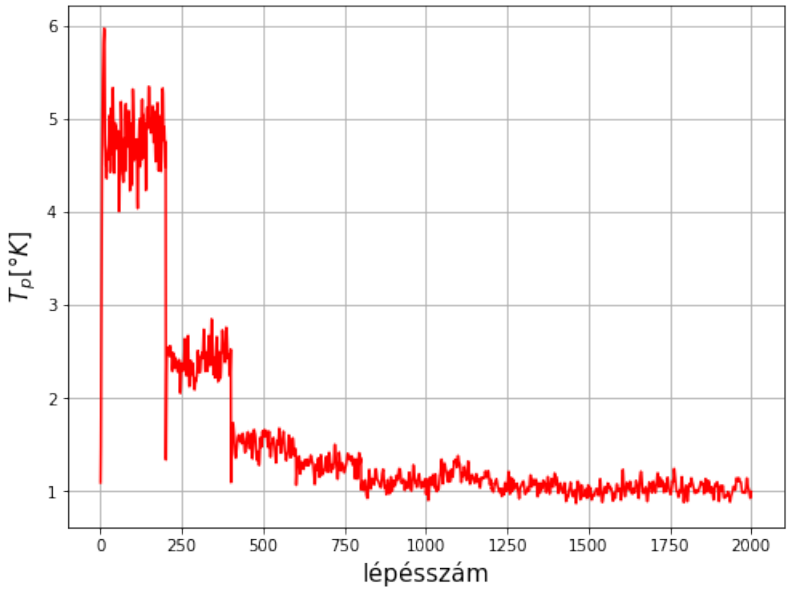


3. ábra. Az md.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, és $n_{max} = 100000$ maximális lépésszám mellett úgy, hogy a teljes rendszer kezdő mérete egy $L = 10m$, és $N = 64$ részecskét tartalmaz.

Tisztán látszik, hogy eddig alábecsültem a tranziens szakasz hosszát - az nyilvánvalóan 20000 lépést tesz ki ezen ábra alapján. Így tehát a kezdeti húszezer lépés kihagyva a számított egyensúlyi hőmérséklet: $T_{egy} = 0.4934K \pm 0.0003K$. A hiba nagyon kicsi, vagyis az adatok szórása is

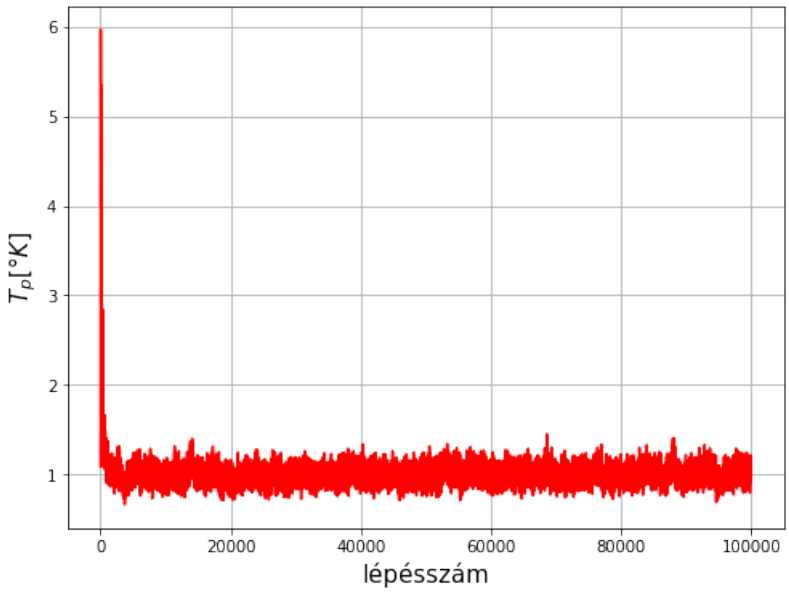
az - ez el is várható egy egyensúlyi helyzetben vett hosszú időátlagtól.

A második program az előzőhöz képest számos változtatást tartalmaz. Egyrészt itt már tömbök helyett pointerekkel dolgozott az alkotó. Bevezetésre kerültek a tranziens szakasz lerövidítését célzó lépések: a részecskék lapcentrált köbös rácsból indulnak, a sebességük pedig Maxwell-Boltzmann eloszlás alapján van generálva (lásd (8)-as,(9)-es, (10)-es képlet, és körülöttük a szöveg). A sebességeket az algoritmus rendszeresen újraskálázza, hogy az elején megadott hőmérsékletnek megfeleljenek, ezzel minimalizálva a numerikus hibákat. Belekerült a periodikus határfeltétel is, amelynek az erők kiszámításánál is fontos szerepe van, hiszen két részecske között mindig a legrövidebb távolságot kell venni konvenció szerint, ami gyakran nem ugyanazon cellában található részecskék távolságát takarja. A pillanatnyi hőmérséklet számítása itt is a (11)-es képlet alapján történik.



4. ábra. Az md2.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, és $n_{max} = 2000$ maximális lépésszám mellett úgy, hogy a teljes rendszer kezdő mérete egy $L = 10m$, és $N = 64$ részecskét tartalmaz.

A program úgy van beállítva, hogy kétszáz lépésenként skálázza újra a sebességeket, ezért láthatóak nagy csökkenések a hőmérsékletben kétszáz, négyszáz és hatszáz lépésnél. A program szemre 1250 lépésnél éri el az egyensúlyi állapotot - hogy erről megbizonyosodjak, lefuttatok egy hosszabb szimulációt.

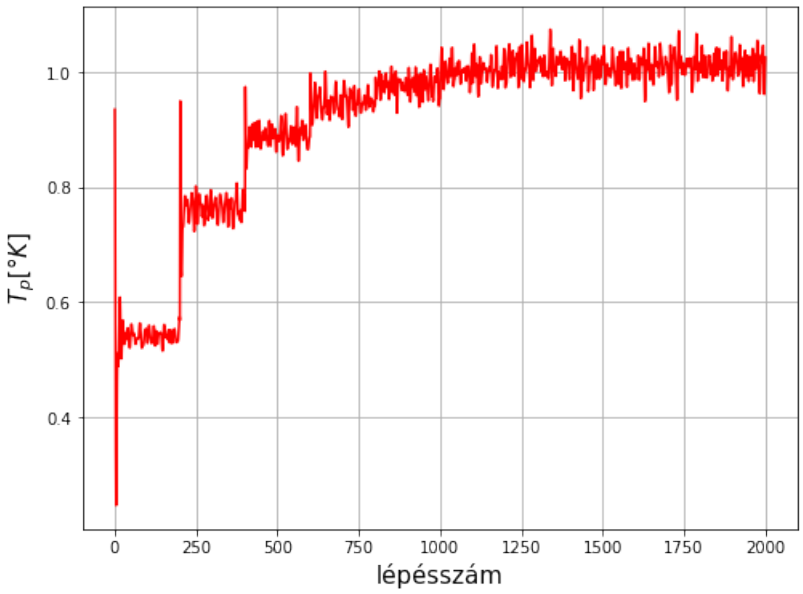


5. ábra. Az md2.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, és $n_{max} = 100000$ maximális lépésszám mellett úgy, hogy a teljes rendszer kezdő mérete egy $L = 10m$, és $N = 64$ részecskét tartalmaz.

Látható, hogy a tranziens rész itt valóban sokkal rövidebb, legalább egy nagyságrenddel kisebb, mint az előző algoritmus eredményeinek esetében, tehát az 1250 megfelelő becslésnek tűnik. Az egyensúlyi hőmérséklet az adatok alapján $T_{egy} = 1.0040K \pm 0.0073K$, vagyis az első három nagyságrendben pontos, ás hibahatáron belül van a beállítotthoz képest. Ezek szerint a hőmérséklet szempontjából is precíz a rendszer.

2.2. A gyorsítást megvalósító algoritmus vizsgálata

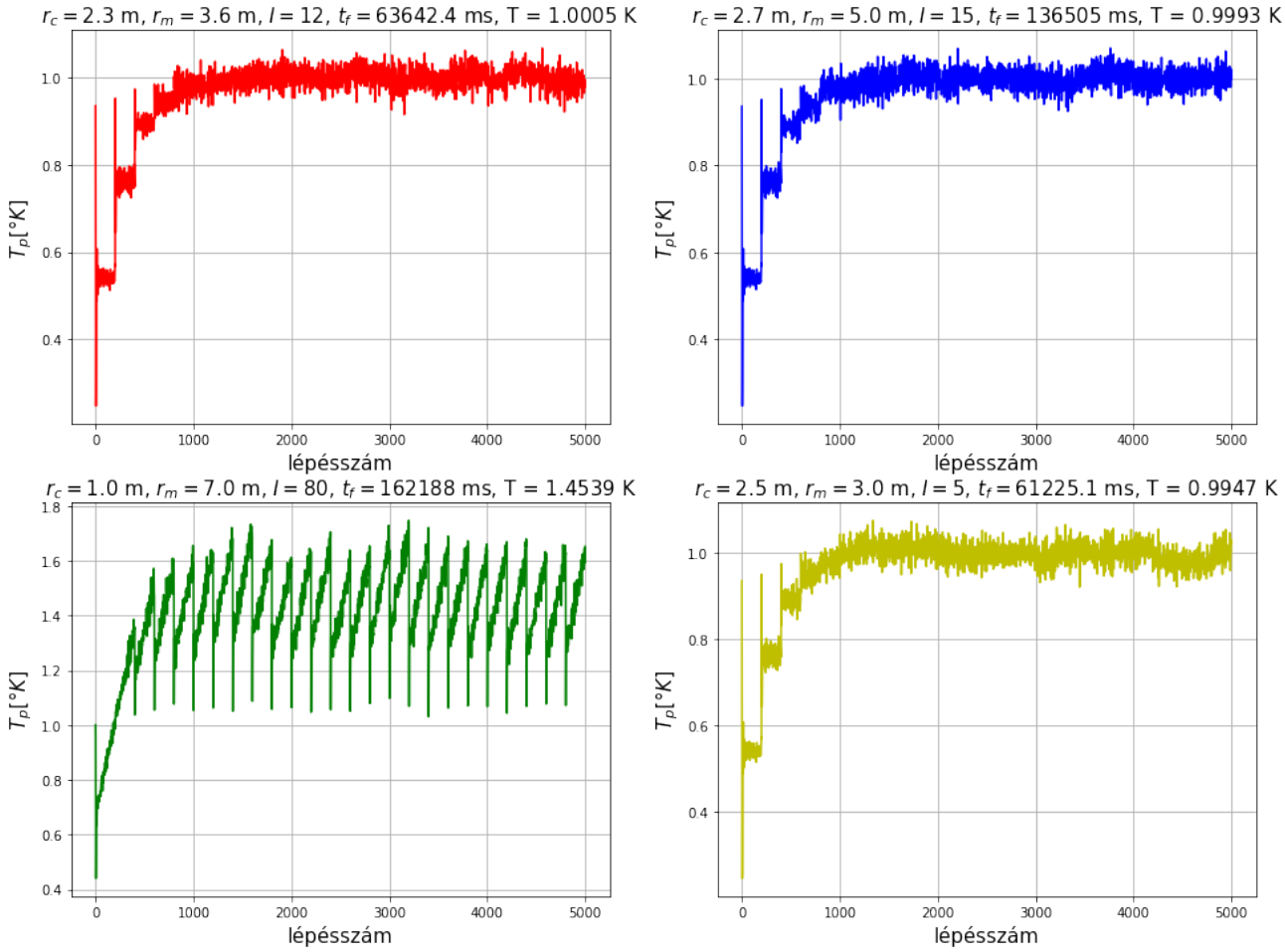
A harmadik algoritmus a másodikhoz képest (tehát az md2.cpp az md3.cpp-hez képest) annyiban más, hogy ez már a harmadik oldal alján részletezett gyorsítási változtatásokat is tartalmazza. Tehát van egy r_{cutoff} és egy r_{max} paraméter, amelyek az úgynevezett szomszédsági listák tartalmát szabályozzák - ezen szomszédsági listák tartalmazzák azon részecskéket, amelyek olyan közel vannak az aktuálisan vizsgált részecskéhez, hogy már jelentős erőhatás van köztük - az összes többi részecskével való kölcsönhatást pedig elhanyagolhatónak lehet tekinteni. A szomszédsági listák megadott lépésszámonként (legyen I) frissülnek.



6. ábra. Az md3.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrája $dt = 0.01s$ lépéshossz, $T = 1K$ beállított hőmérséklet és $n_{max} = 2000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$, és $N = 864$ részecskét tartalmaz. A futási idő $t_f = 23497.1ms$ volt.

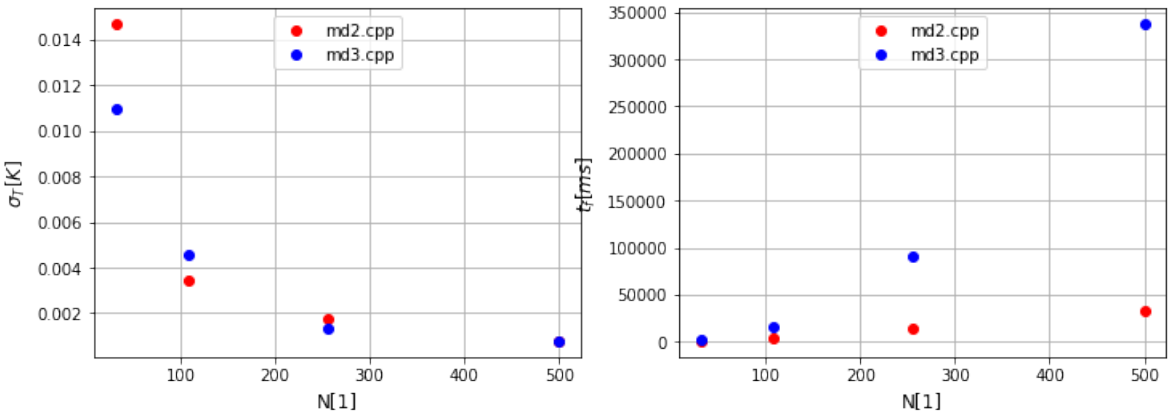
Összességében nagyon hasonlít az eredmény az md2.cpp hőmérséklet-lépésszám ábrájára, nem véletlenül, hiszen a változtatások csak a futási idő gyorsítását célozták meg.

Az imént használt gyorsítási paraméterek ajánlások az algoritmus kitalálójától, L. Verlet-től. De megpróbálkozhatunk más paraméterekkel is, ahogy az a következő ábrákon is látható.



7. ábra. Az md3.cpp program által megvalósított velocity-Verlet-algoritmus hőmérséklet-lépésszám ábrái $dt = 0.01s$ lépéshossz, $T = 1K$ beállított hőmérséklet és $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a rendszer és $N = 864$ részecskét tartalmaz. Az ábrák tetején látható egyensúlyi hőmérséklet kalkulációjakor csak az utolsó háromezer értéket vettem figyelembe.

Az első két szimuláció esetén nem látszódnak nagy különbségek a Verlet által ajánlott paraméterek szerint futtatottól, csupán a futási idő nőtt meg. Azonban a harmadik esetben a futási idő drasztikusan nagy lett, és már az egyensúlyi hőmérséklet is pontatlanná vált, az ábrán pedig egyfajta különös oszcilláció látszik. Ez azzal magyarázható, hogy ez esetben nyolcad olyan gyakran lettek kiszámítva a részecskékre ható erők, és a ez a rendszer összeomlását, az energia elszabadulását okozná, ha kétszáz lépésenként nem lennének újraskálázva a sebességek. A negyedik szimuláció esetében csupán annyi történt, hogy r_{max}-ot kicsit kisebbnek választottam, és a javasoltnál gyakrabban frissítettem a szomszédsági listát - ez azt okozta, hogy a hőmérséklet az optimálisnál pontatlanabb lett, és nagyobb lett a futási idő.

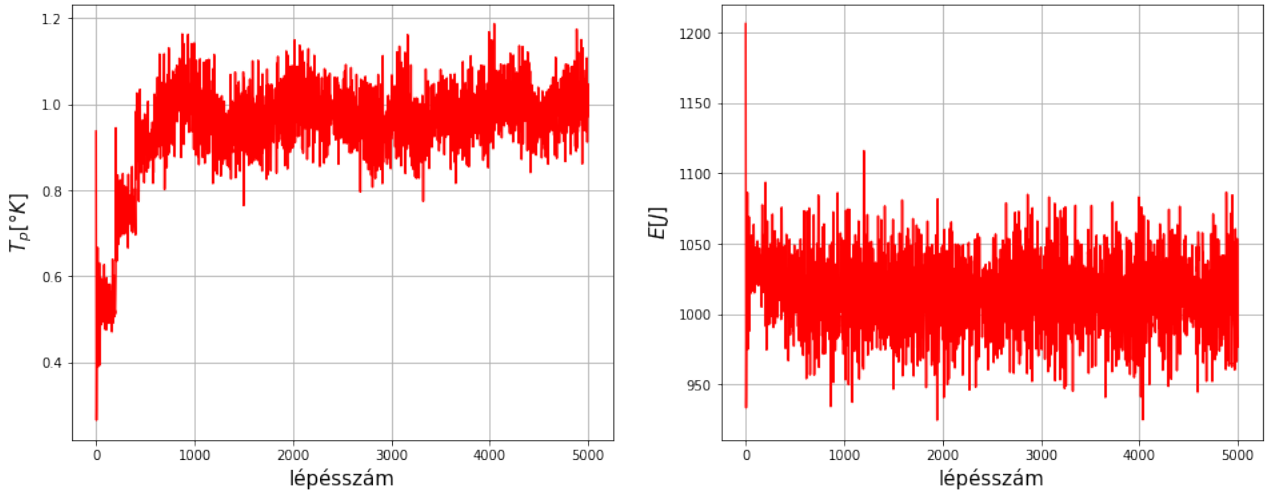


8. ábra. Az md2.cpp és az md3.cpp programok pontosságának és futási idejének összehasonlítása $dt = 0.01s$ lépéshossz, $T = 1K$ beállított hőmérséklet és $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$ voltak. σ_T az egyensúlyban mért pillanatnyi hőmérsékletek szórását jelöli. A hőmérsékletek mérésénél a tranziens szakaszt, vagyis az első kétezer lépést nem vettem figyelembe.

Az ábra alapján a hőmérsékletmérés pontossága hasonló a két algoritmus esetén a betáplált részecskeszámok esetén. Ugyanakkor sikerült igazolni, hogy a gyorsításokkal az eredetileg N^3-ös időfüggést lineárissá lehet redukálni.

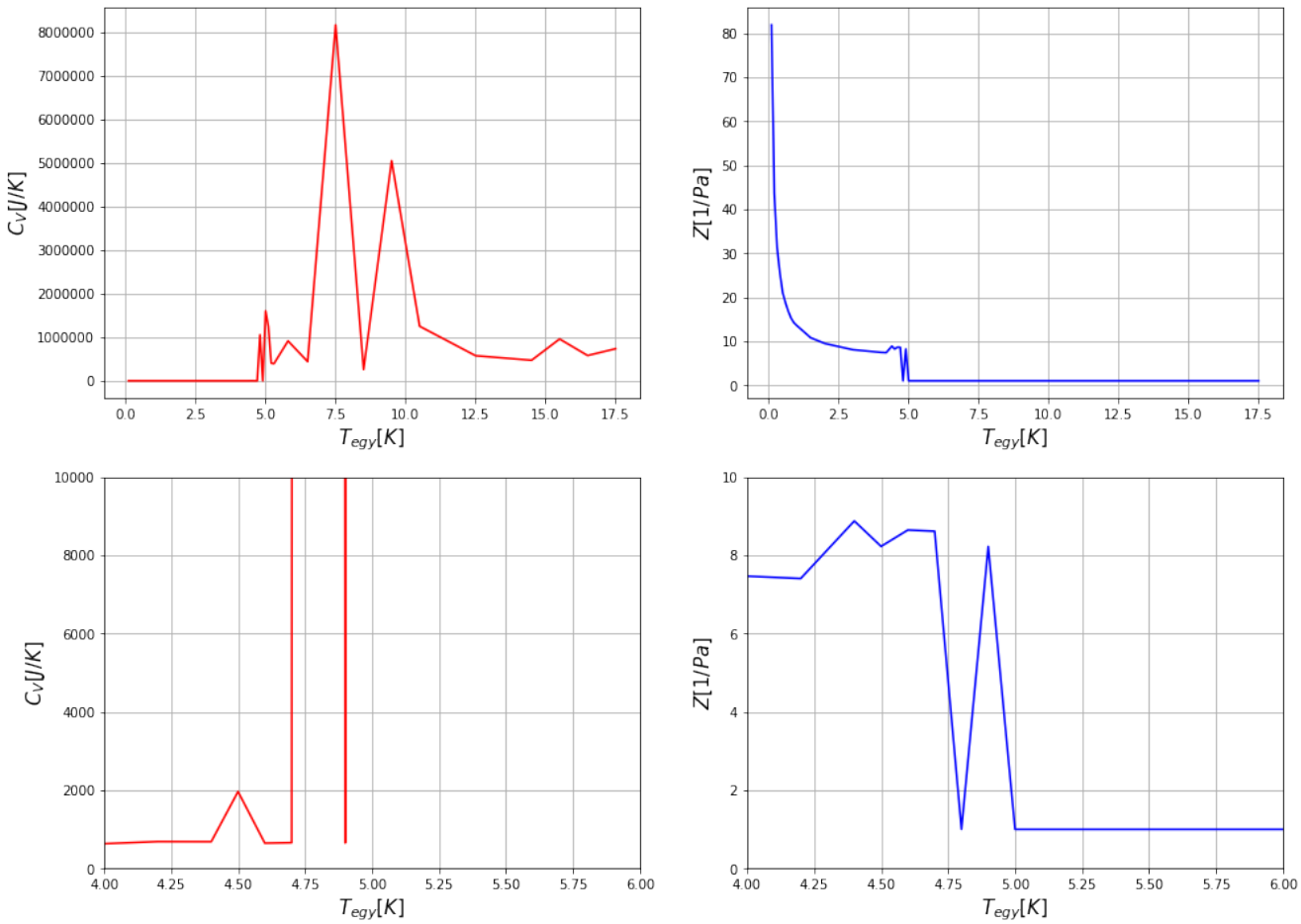
2.3. Termodinamikai mennyiségek meghatározása

Implementáltam a kódba a bevezető fejezetben említett termodinamikai mennyiségeket az ottani képletek ((12)-(15)) alapján.



9. ábra. Az md3.cpp program eredményei $dt = 0.01s$ lépéshossz, $N = 109$ részecskeszám, $T = 1K$ beállított hőmérséklet és $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$ voltak. Az eredmények: $C_V = 656.203\frac{J}{K}$, $P = 17.3226Pa$, $Z = 12.2328\frac{1}{Pa}$.

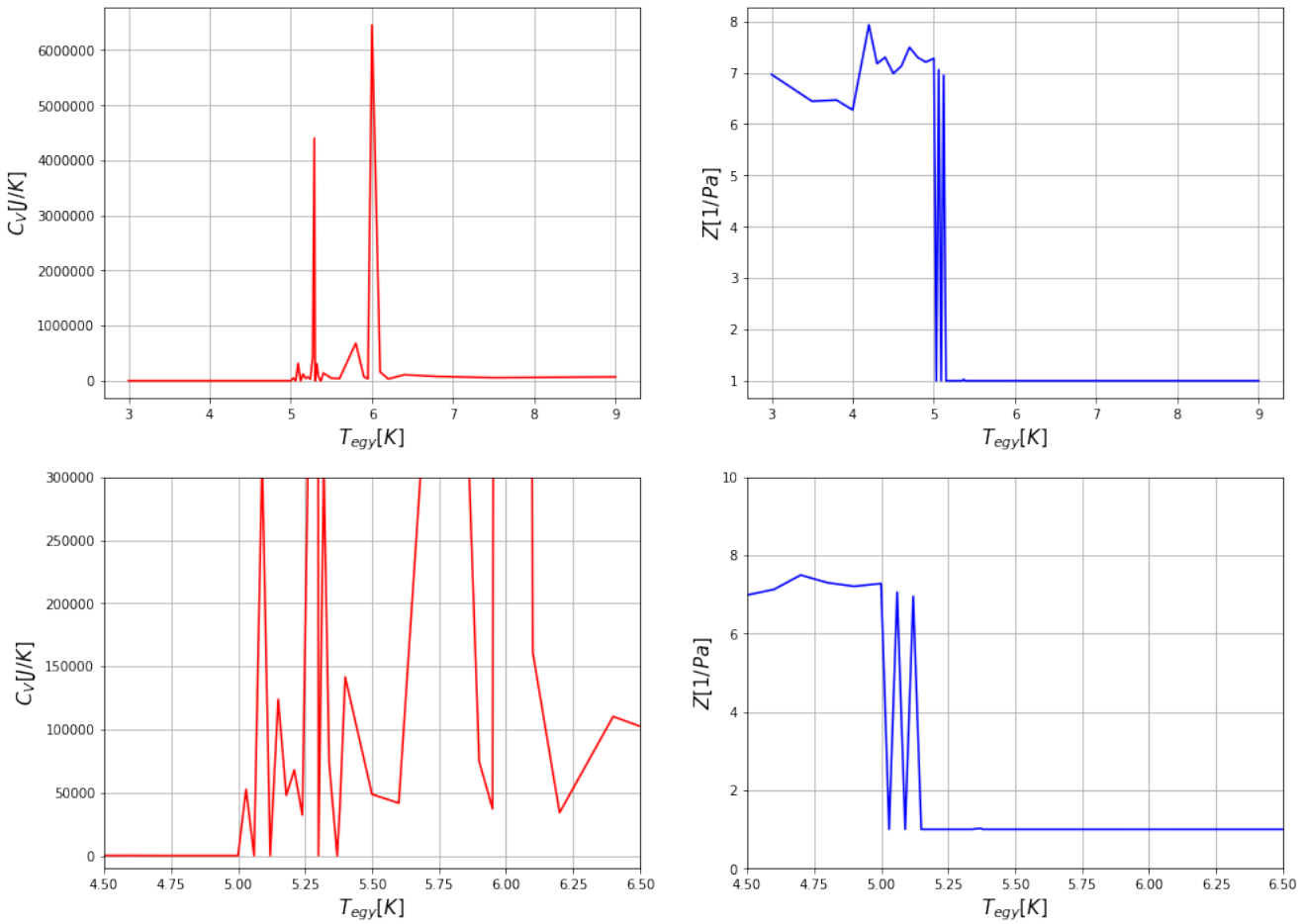
A teljes energia láthatóan megmarad, bár enyhén fluktuál, de ez várható is volt, így tehát valószínűleg megfelelően van számítva. A többi mennyiség esetén időátlagokról beszélünk, és mivel át lettek skálázva az értékek még a számolások előtt, nem tudom összehasonlítani ilyenformán a valós értékekkel a kapottakat, így ellenőrizni sem tudom őket.



10. ábra. Az md3.cpp program eredményei $dt = 0.01s$ lépéshossz, $N = 108$ részecskeszám, $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$ voltak.

Az ábrák alapján valahol $4.6K$ és $4.9K$ között lehetnek a fázisátalakulások. Ez egybevág a szakirodalommal, miszerint az argon olvadáspontja $83.80K$-nél, a forráspontja pedig $87.30K$-nél van (megint nem szabad elfelejteni, hogy a mi eredményeink mértékegysége nem ugyanaz a K, mint az SI-ben). Érdemes lehet tehát más részecskeszámmal részletesebben feltérképezni ezen tartományt.

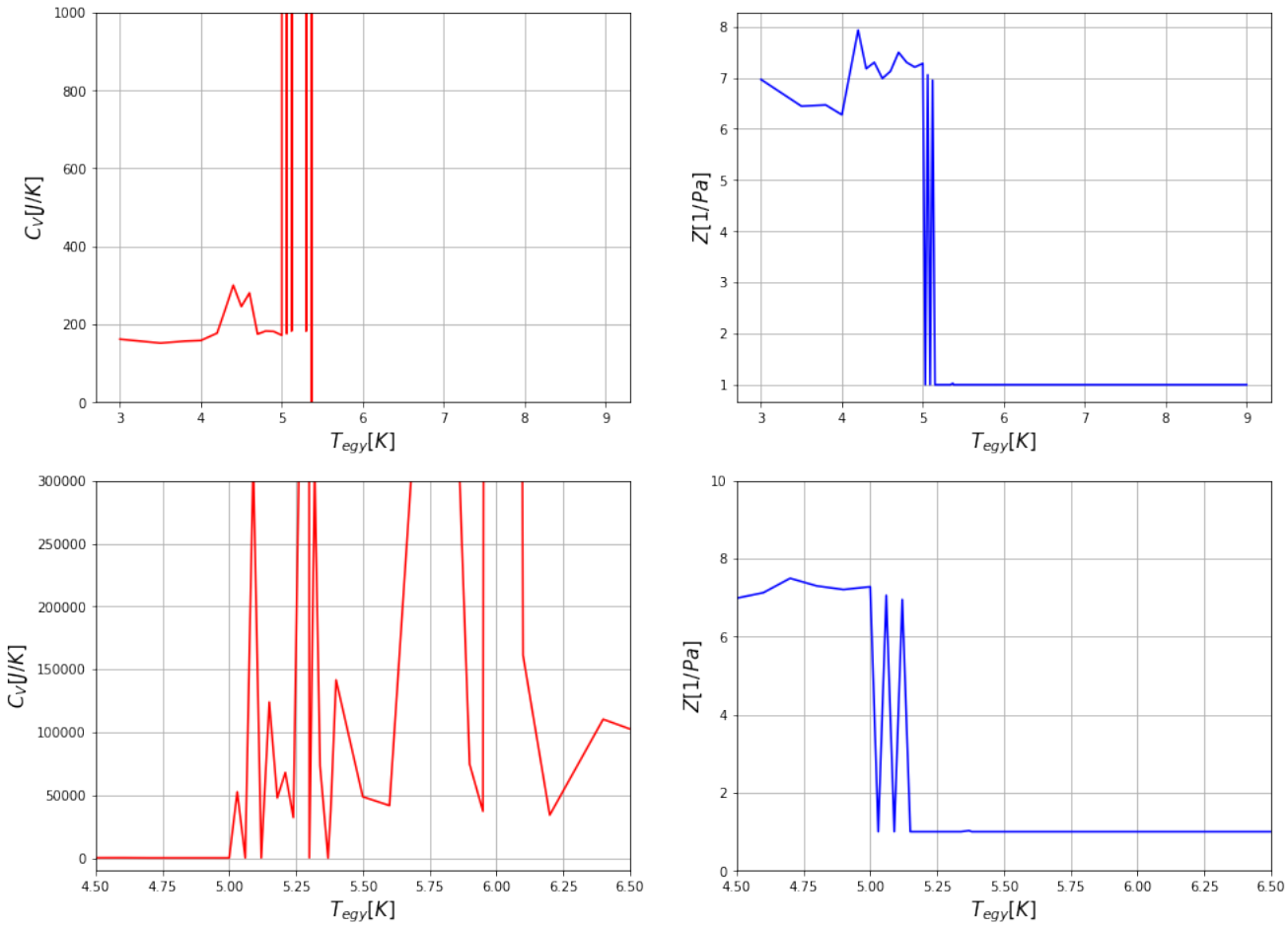
Az mindenesetre ezen az ábrán is látszik, hogy $5K$ felett a rendszer az ideális gáz állapotában van (mivel $Z = 1$), tehát valószínűleg ott légnemű.



11. ábra. Az md3.cpp program eredményei $dt = 0.01s$ lépéshossz, $N = 32$ részecskeszám, $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$ voltak.

A részecskeszám harmadára csökkentésének következményeképp a kompresszibilitás egy tized fokkal nagyobb hőmérsékleten vág le egyre, vagyis ez alapján ha csökkentem a részecskeszámot, nőnek a fázisátalakulási hőmérsékletek, míg ha növelem a részecskeszámot, csökkennek. Tapasztalataim szerint a hőkapacitás viszont arányos (ha nem is egyenesen) a részecskeszámmal.

Az látszik az ábrákon, hogy $4 - 4.2K$-nél növekedésnek indul a kompresszibilitás, holott addig végig csökkent - lehet, hogy itt látható egy fázisátalakulás szilárdból folyékonyra. Kicsit igazítok az ábrákon, hogy jobban kivehetők legyenek a vizsgált változások.



12. ábra. Az md3.cpp program eredményei $dt = 0.01s$ lépéshossz, $N = 32$ részecskeszám, $n_{max} = 5000$ maximális lépésszám mellett úgy, hogy $\rho = 1.2$ részecske indul egy rácscellából (amely rácscellák egy kockát formáznak együttesen), a gyorsítási paraméterek $r_{cutoff} = 2.5m$ ($r_0 = 1m$), $r_{max} = 3.3m$, $I = 10$ voltak.

Látható, hogy $4.2K$-nél a hőkapacitás is megnövekedett. Az biztos, hogy $5.1K$ felett nincs több fázisátalakulás, mert a kompresszibilitási együttható ott változatlanul egy, ami arra utal, hogy ideális gáz állapotában van a rendszer. Vagyis azt szűrhetjük le, hogy $T = 4.2K$-nél megy át az anyag folyékony halmazállapotba, és $T = 5.1K$-nél gáz halmazállapotba. Ez egyúttal azt is jelenti, hogy $N = 32$ részecskeszám esetén egymáshoz és a kezdőponthoz viszonyítva kétszer akkora távolság van a fázisátalakulási hőmérsékletek között, mint a valódi argon esetében.

3. Összefoglalás

Bemutatásra kerültek a molekuladinamikai szimulációkra kitalált Verlet és velocity-Verlet algoritmusok. Az utóbbi különböző megvalósításainak eredményeit megvizsgáltuk, a programokat egymással is összehasonlítva - az első programhoz (md.cpp) képest a másodikban már a kezdeti feltételek pontosabb beállítását is implementálták a sebességek rendszeres átskálázásával egyetemben. Ennek következményeit, vagyis a tranziens szakasz lerövidülését, és a nagyobb pontosságot szemléltettük ábrákkal is.

Az md3.cpp program már a gyorsítást célzó változtatásokat is tartalmazta. Ennek eredményeit szemléltettük, és mind futási idő, mind pontosság tekintetében összehasonlítottuk az előző programmal. Azt találtuk, hogy az egyensúlyi hőmérséklet pontossága megközelítőleg megegyezik, ellentétben a futási idővel, amely szemlátomást jócskán lecsökkenthető nagy részecskeszámok esetén az eszközölt változtatásokkal. Kísérleteztünk más gyorsítási paraméterekkel is, és azt találtuk, hogy valószínűleg tényleg az ajánlottak a legmegfelelőbbek.

A legutolsó programot továbbfejlesztettük úgy, hogy különböző termodinamikai mennyiségeket is kiszámoljon. Ezáltal meg arra a következtetésre jutottunk, hogy $N = 32$ részecske esetén egymáshoz és a kezdőponthoz viszonyítva kétszer akkora távolság van a fázisátalakulási hőmérsékletek között, mint a valódi argon esetében. Mindezeken túl az is kiderült, hogy a rendszer gáz halmazállapotban a kompresszibilitás szempontjából ideális gázként viselkedik légnemű halmazállapotban. Tapasztalataink szerint a részecskeszám növelése valamelyest csökkenti a fázisátalakulási hőmérsékleteket, míg az állandó térfogaton mért hőkapacitás valamelyen arányosságot mutat a részecskeszámmal (ha nem is feltétlenül egyenest).