Molekula dinamika

Számítógépes szimulációk ff1n4i11/1

Csabai István

ELTE
Komplex Rendszerek Fizikája Tanszék
5.102

Email: csabaiΘcomplex.elte.hu

2009 tavasz

Nagy számú részecske szimulációja

- A molekula dinamika (Molecular dynamics, (MD)) "részecskék" mikroszkopikus dinamikájának követésével foglalkozik.
- Makroszkopikus rendszerben a részecskék száma 6×10^{23} nagyságrendű. A jelenlegi számítógépekkel nem tudjuk ennyinek a mikroszkopikus dinamikáját követni, de lehetséges *elég* sok részecskét szimulálni, amelyekkel már termodinamikus tulajdonságok vizsgálhatóak.
- Cél, hogy minél több részecskét, minél pontosabban szimuláljunk, ehhez közelítéseket vezetünk be. A részecskék között páronkénti kölcsönhatások vannak, tehát N részecske esetén elvileg N^2 erőhatást kell szimulációs lépésenként kiértékelni.
- Ha az erő elég gyorsan lecseng (pl. a molekulák közt tipikusan fellépő Van der Waals erő), akkor a távoli erők nagyobb térrészekre kiátlagolhatóak, nagyon távol pedig elhanyagolhatóak, így a módszer jelentősen gyorsítható. Az ún. hosszú hatótávolságú erőknél (pl. a gravitáció) az elhanyagolás nem lehetséges, más közelítéseket kell alkalmazni, ezeket a molekula dinamikától megkülönböztetendő, N-test szimulációnak (N-body) nevezik.

Molekula dinamika algoritmus vázlata

- Legyen N (nagy számú) atomunk, melyeknek ismerjük kezdő sebességüket és pozíciójukat
- A Newton törvények értelmében a részecskék közt páronként számoljuk ki az erőt, és léptessük a részecskéket
- Figyeljük, hogy a rendszer a kezdeti tranziens után (nem egyensúlybeli kezdeti feltételek miatt) termodinamikai egyensúlyba került-e. Az egynsúly esetében teljesül az ekvipartíció:

$$\langle K \rangle = \left\langle \frac{1}{2} m v^2 \right\rangle = \frac{3}{2} kT$$

  ahol $k = 1.38 \times 10^{-23} J/K$; a Boltzmann állandó. A szimuláció során $\langle K \rangle$ mutatja, hogy egyensúlyban van-e a rendszer.
- Ha a rendszer elérte az egyensúlyt, mérhetőek a termodinamikai változók: hőmérséklet (T), nyomás (p), hőkapacitás (C_V), stb.

Közelítések

- Elvileg az atomok belső szerkezetét is figyelembe vevő kvantummechanikát kellene használni. De pl. $T = 300K$ argon gázban az egy részecskére jutó mozgási energia

$$\frac{3}{2}kT \cong 6.2 \times 10^{-21}\mathrm{J} = 0.039eV$$

sokkal kisebb mint az elektronok gerjesztési energiája ($11.6\mathrm{eV}$). Valamint az $m = 6.69 \times 10^{-26}\mathrm{kg}$ tömegű atomok de Brolglie hullámhossza

$$\frac{2\pi\hbar}{p} = \frac{2\pi\hbar}{\sqrt{3mkT}} \cong 2.2 \times 10^{-11}\mathrm{m},$$

jóval kisebb mint az atom effektív mérete ($r_0 = 3.4 \times 10^{-10}\mathrm{m}$), így az atomok közötti tipikus távolságnál is. A nemesgázoknak (pl. az argon) zártak az elektronhéjai, így szimulációjuk során a fentiek figyelembe vételével, a klasszikus részecske közelítés elfogadható eredményre vezet.

- Megviszgálva az időskálákat, probléma lehet a termodinamikai egynsúly elérése is. A szimulációs lépéshossznak kisebbnek kell lenni, mint az atomok közötti átlagos távolság megtételéhez szükséges idő, az atomok átlagos sebességével, ami tipikusan:

$$\frac{r_0}{\sqrt{3kT/m}} \cong 7.9 \times 10^{-13}\mathrm{s},$$

azaz kevesebb mint 1 picosecundum. Ha $N = 10^3$ atomra minden lépésben $N(N-1)/2$ erőt kell kiszámolni, akkor egy gyors gépen is órákig tarthat akár néhány nanosecundum szimulációja! Ezért fontos a kezdeti feltételeknek a termodinamikai egyensúlyhoz minél közelebbi beállítása.

A Verlet algoritmus

- A bolygómozgásnál már megismert módon most is differenciálegyenlet rendszert kell integrálni. Természetes választás lenne, az Euler-, vagy az annál pontosabb Runge-Kutta módszer.
- A legelterjedtebb MD differenciálegyenlet léptető algoritmus mégis inkább a Verlet algoritmus.

- Jelölje $\mathbf{R}(t) = (r_1, \ldots, r_N)$ a részecskék koordinátáit, hasonlóan $\mathbf{V}(t)$ a sebességeket, $\mathbf{A}(t)$ pedig a gyorsulásokat. A Verlet algoritmus egy lépése:

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

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

- Előnyök:
  - Gyorsabb mint a Runge-Kutta, egy lépés során, csak egyszer kell kiszámolni a gyorsulásokat
  - Majdnem olyan pontos, mint a 4-ed rendű Runge-Kutta, ahol a hiba $\mathcal{O}(\tau^5)$ rendű
  - Jól megőrzi az energiát, ami nagyon fontos a sokrészecske dinamikánál
  - Időtükrözésre nem változik, ami fontos a részletes egyensúly (ergodicitás) feltétele miatt

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

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

- Hátrányok:
  - *Három pontos* rekurzió, azaz két előző lépést használ, nem indítható egy kezdeti feltételből
  - A sebesség hibája $\mathcal{O}(\tau^2)$, viszont ha az erő nem függ a sebességtől, akkor ez nem rontja a pozíciók pontosságát.
  - A sebesség és a pozíció nem egyszerre frissítődik, a sebesség "le van maradva"
- A problémák egy részén segít a *velocity-Verlet* algoritmus:

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

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

- Ez $\mathbf{R}$-ben csak $\mathcal{O}(\tau^3)$ rendben pontos, de nem használ két előző lépést, egyszerre frissíti a koordinátákat és sebességeket, valamint a sebesség is pontosabb.

A 2 dimenziós Lennard-Jones rendszer

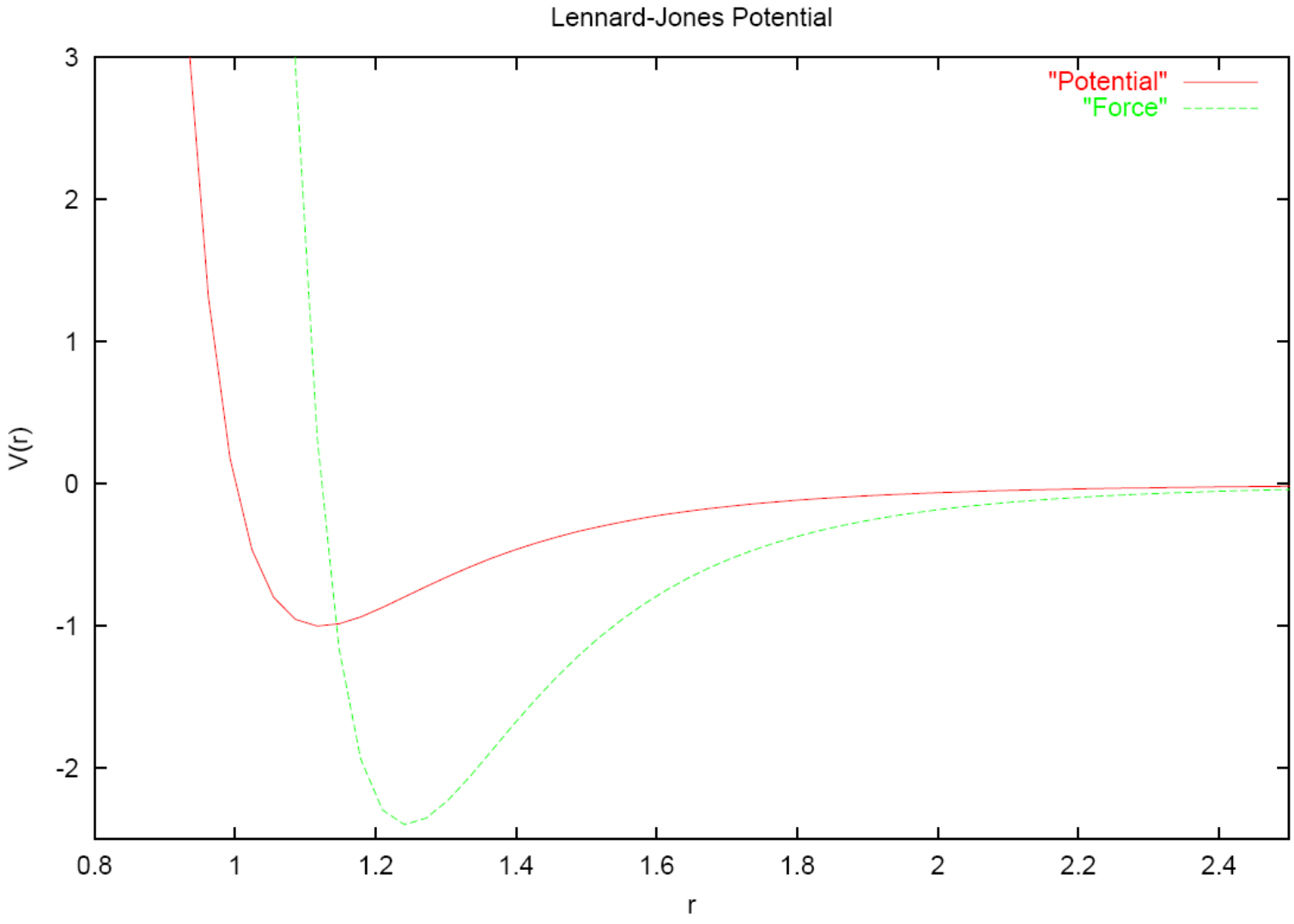
- Modelezzük N darab "2 dimenziós argon atom" viselkedését egy négyzetben. Azért, hogy a határokon ne kelljen speciális feltételekkel foglalkozni (fallal ütközések, nyílt rendszernél részecskeszám megmaradás), alkalmazzunk periodikus határfeltételt: azonosítsuk a szemközti oldalakat (tórusz toplógia), így egy "határtalan" rendszert kapunk.
- A részecskék közötti kölcsönhatást a van der Waals közelítésben a Lennard-Jones potenciál írja le:

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

  ahol r az atomok középpontja közötti távolság. Argonra $V_0 = 1.65 \times 10^{-21} J$ azaz $V_0/k_B = 119.8$K és $r_0 = 3.41 \times 10^{-10}$m.
- A potenciál gradienséből számolható erő:

$$\mathbf{F}(\mathbf{r}) = -\vec{\nabla}(r) = \frac{24V_0}{r^2} \left[2 \left(\frac{r_0}{r} \right)^{12} - \left(\frac{r_0}{r} \right)^{6} \right] \mathbf{r}$$

A Lennard-Jones potenciál $r_0 = 1$ és $V_0 = 1$ esetén:



- A potenciál $r = r_0$-nál 0 és $r = 2^{1/6} r_0$-ben veszi fel minimumát, ami $-V_0$
- At erő erősen taszító kis távolságokon a zárt elektronhéjak Pauli-féle kizárási "taszítása" miatt,
- nagy távolságokon pedig gyengén vonzó, az indukált elektromos dipólerők (van der Waals kölcsönhatás) miatt
- A rendszer energiája:

$$E = \sum_{i=1}^{N} \frac{1}{2} m v_i^2 + \sum_{\langle ij \rangle} V(r_{ij}),$$

  ahol r_{ij} az $\langle ij \rangle$ atompár közötti távolság.
- Egyensúlyban a sebességek eloszlásának a Maxwell-Boltzmann eloszlást kell követni, ami 2 dimenzióban:

$$\mathcal{P}(v)dv = \left(\frac{m}{kT}\right) e^{-\frac{mv^2}{2kT}} v dv.$$

2D Lennard-Jones szimuláció

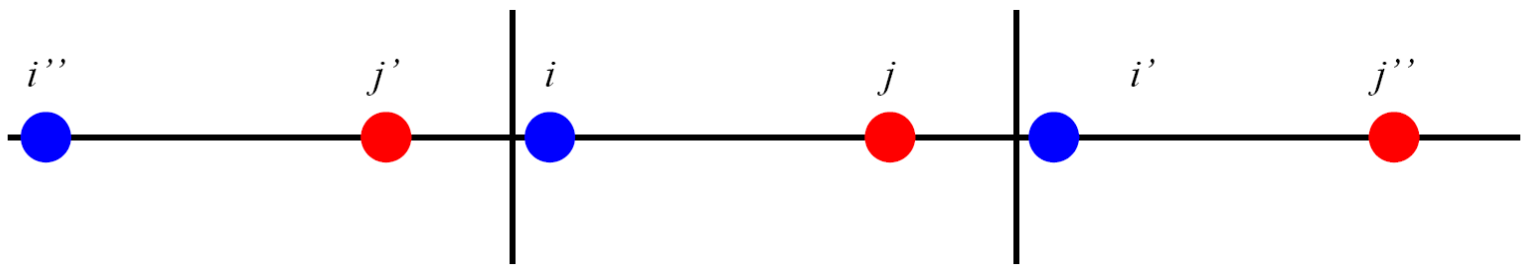
- A szimuláció során a valódi SI értékekkel numerikus szempontból nem praktikus számolni, ezért a változókat átskálázzuk (hasonló mint a realtivitáselméletben szokásos $c = 1$ mértékegységrendszer.) Úgy választjuk meg a skálát, hogy az argon atom tömege legyen a tömegegység $m = 1$, $r_0 = 1$ és $V_0 = 1$. A rendszer karakterisztikus idejét

$$\tau_0 = \sqrt{\frac{m r_0^2}{V_0}} = 2.17 \times 10^{-12}\mathrm{s},$$

  ami pár picosecundum, szintén skálázzuk át, $\tau_0 = 1$. Vegyünk például $N = 16$ atomot, az időlépés pedig legyen a karakterisztikus idő századrésze, $\tau = 0.01$
- Az egy részecskére ható erő kiszámolásához, össze kell adni az összes többi részecske hatását:

$$\mathbf{F}_i = \sum_{\substack{j=1 \\ j \neq i}} \mathbf{F}_j \text{ hat } i\text{−re}$$

- Az algoritmus fontos új része a periodikus határfeltétel kezelése. Mivel az elemi L^3 térfogatban lévő részecskék periodikusan ismétlődnek, meg kell oldani, hogy ne legyen végtelen sok kölcsönhatás. A részecske saját képeinek hatása kioltja egymást, de más részecskék többszörösen több irányból hathatnának, ami olyan korrelációkat hozna be ami nem lenne fizikai.
- A legközelebbi kép nem mindig az aktuális cellában van, lehet annak valamelyik első szomszédjában is. (Belátható, hogy távolabb nem lehet.) Például, lehet, hogy $r_{ij} = |\mathbf{r}_i - \mathbf{r}_i|$ nagyobb mint egy szomszédos cellában lévő $r_{ij'} = |\mathbf{r}_i - \mathbf{r'}_i|$



- Hogy egy adott részecskére minden más részecske csak egyszeresen hasson, válasszuk mindig a legközelebbi megjelenését (legnagyobb erőhatást), a fenti 1D példán i-hez a j'-t a j helyett.
- Gondoskodni kell a kódban arról is, hogy ha egy részecske pl. jobb oldalon elhagyja a négyzetet, akkor bal oldalon visszakerüljön, stb.

- Nem nyilvánvaló a kezdőfeltételek beállítása se. Ha nem realisztikus elrendezést és sebességeket állítunk be, akkor sok időbe telik míg a kezdeti tranziens után egyensúlyba kerül a rendszer.
  - Kondenzált állapotban az atomok általában lapcentrált köbös rács pontjaiban helyezkednek el, ez minimalizálja a potenciális energiát. Az egyszerű változatban e helyett egyszerű köbös rácsot használunk.
  - A sebességeknek a Maxwell-Boltzmann eloszlást kellene követni, e helyett egyelőre egy adott intervallumban eloszló egyenletes véletlen értékekről indítjuk a sebességeket.

Pillanatnyi hőmérséklet

A Lennard-Jones erő konzervatív, rendszer teljes energiája megmarad. Termikus egyensúlyban teljesül az ekvipartíció tétele, amikor is a kinetikus energiából származtatható a hőmérséklet:

$$3(N-1) \times \frac{1}{2} k_B T = \left\langle \frac{m}{2} \sum_{i=1}^{N} v_i^2 \right\rangle .$$

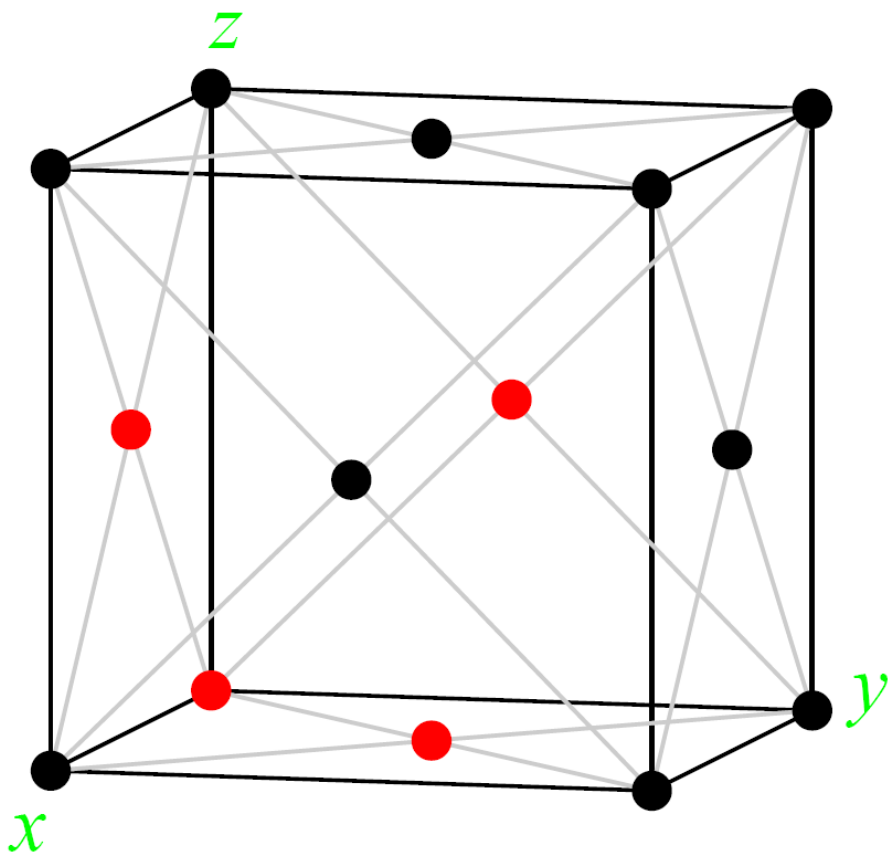
$\langle \ldots \rangle$ jelöli a termikus sokaság átlagot. $3(N-1)$, a belső szabadsági fokok száma, a tömegközéppont mozgása nem járul hozzá a termikus energiához, ha az átlag sebesség $\bar{v} \neq 0$, akkor igazából v_i^2 helyett $(v_i - \bar{v})^2$ szerepel a kifejezésben. Nem-egyensúlyi helyzetben is lemérhetjük ezt a mennyiséget, ekkor a nem-egyensúlyi (nem igaz az ekvipartíció) jelleg megkülönböztetéseként *pillanatnyi hőmérsékletnek* nevezhetjük.

Programok

- A md.cpp program a legegyszerűbb 3 dimenziós Lennard-Jones szimuláció. Nem használ periodikus határfeltételeket, a velocity-Verlet algoritmussal léptet, és a pillanatnyi hőmérsékletet egy fájlba menti ki.
- A md-gl.cpp program a velocity-Verlet algoritmust használja periodikus határfeltételek mellett. Folyamatosan méri a sebességhisztogramot, és összeveti a Maxwell-Boltzmann eloszlással. Az egérgombok segítségével indítható/állítható a szimuláció, illetve növelhető/csökkenthető a hőmérséklet.

Fejlesztések a kódon

- A 3 dimenziós kódban is alkalmazzuk a periodikus határfeltételt!
- Alacsony hőmérsékleten, amikor a potenciális energia minimuma körül van a rendszer, a rendszer a lapcentrált köbös állapotban éri el az energiaminimumot. Az elemi cella 4 atomot tartalmaz (piros):



- Akkor tudunk jól kitölteni egy kockát, ha az atomok száma $N = 4M^3$, $M = 1, 2, 3, \ldots$, pl. $32 = 4 \times 2^3$, $108 = 4 \times 3^3$, $256, 500, 864, \ldots$
- Az 1 méretű elemi cellában lévő atomok pozíciói

$$(0, 0, 0) \quad (0.5, 0.5, 0) \quad (0.5, 0, 0.5) \quad (0, 0.5, 0.5)$$

de hogy ne a határon legyenek toljuk el $(0.5, 0.5, 0.5)$-tel, így a kódban:

```
// 4 atomic positions in fcc unit cell
double xCell[4] = {0.25, 0.75, 0.75, 0.25};
double yCell[4] = {0.25, 0.75, 0.25, 0.75};
double zCell[4] = {0.25, 0.25, 0.75, 0.75};
```

Kezdeti sebességek Maxwell-Boltzmann eloszlás szerint

- T hőmérsékleten a Maxwell-Boltzmann eloszlás

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

  vagyis minden sebességkomponens 0 várható értékű, $\sqrt{T}$-vel arányos szórású Gauss eloszlással írható le.

- A kódban használt a *Numerical Recipes*-ből átvett `gasdev()` függvény 0 várható értékű, 1 szórású véletlen számokat állít elő a gépi egyenletes eloszlásokból a Box-Muller algoritmus segítségével.

- Az így generált kezdősebességek átlaga 0-hoz közeli lesz, de nem egzaktul 0, a tömegközéppont sebességével korrigálni kell a kezdőértékeket: $\mathbf{v}_i := \mathbf{v}_i - \mathbf{v}_{CM}$, ahol a tömegközépponti sebesség:

$$\mathbf{v}_{CM} = \frac{\sum_{i=1}^{N} m\mathbf{v}_i}{\sum_{i=1}^{N} m}$$

- Ezek után az 1-re normált szórású sebességeket a kívánt sebességre kell átskálázni hogy a kívánt T hőmérsékletet kapjuk:

$$\mathbf{v}_i \to \lambda \mathbf{v}_i$$

$$\lambda = \sqrt{\frac{2(N-1)k_B T}{\sum_{i=1}^{N} m v_i^2}}$$

- A md2.cpp program tartalmazza a fenti javításokat, figyeli a pillanatnyi hőmérsékletet, és amennyiben nem felel meg a megjelölt értéknek, újra átskálázza a sebességeket.

Gyorsítási lehetőségek

- A program legidőigényesebb része a páronkénti erők kiszámítása, $N(N-1)/2$ pár, azaz $\mathcal{O}(N^2)$-tel skálázik a futási idő.
- L. Verlet egy cikkében (*Phys. Rev.* **159**, 98 (1967)) két gyorsítást is bevezetett:
- **A potenciál levágása:** Mivel a Lennard-Jones erő rövid hatótávú, nagysága $r > r_0$ után gyorsan csökken, így bevezethető egy *levágási távolság* r_{cutoff}, amin túl praktikusan 0-nak tekinthető. Így, ha az r_{cutoff} távolságon belül K részecske van és $N = n \times K$, akkor a skálázás teszőleges n-re $\mathcal{O}(n \times K^2)$ rendű. Máshogy megfogalmazva ha a sűrűséget állandóan tartjuk akkor n-ben nem négyzetes, hanem lineáris a skálázás. Nyilván akkor éri ez meg, ha $r_{cutoff} << L$ (L a rendszer mérete).

- **Szomszéd-lista:** A levágás érvénysítéséhez tudni kell, melyek azok a szomszédok, amelyekre $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j| < r_{cutoff}$. Mivel minden részecske mozog, ennek kiértékelése is $\mathcal{O}(N^2)$ feladatnak tűnik. Ha azonban figyelembe vesszük, hogy egy-egy lépésben minden részecske keveset mozdul el, és választunk egy megfelelő $L >> r_{max} > r_{cutoff}$ távolságot, amelynél jobban néhány lépésen belül nem távolodnak el a részecskék. Így elég azon belül nyilvántartani a szomszédsági listát, és csak néha frissíteni a kölcsönható párok listáját. Verlet $r_{cutoff} = 2.5r_0$, $r_{max} = 3.2r_0$ értékeket javasolta, és 10 lépésenként frissítette a szomszédsági listát. (A frissítés gyakorisága és r_{max} közötti összefüggés a sebességek valószínűségeloszlásából becsülhető).
- Verlet így 10-szeres sebességnövekedést ért el a pontosság jelentős romlása nélkül.
- A levágás kisebb hibákat behoz a szimulációba és elrontja az energia megmaradást. A hiba korrigálható a potenciál módosításával:

$$U_{corr}(r) = U(r) - \frac{d}{dr}U(r_{cutoff})(r - r_{cutoff})$$

(Nincs a kiadott kódokban implementálva.)

Gyorsított programok

- A md3.cpp program tartalmazza a fenti Verlet-féle közelítésesket, gyorsítást.
- A md3-gl.cpp program a fenti kód 3 dimenziós OpenGL változata.

Termodinamikai mennyiségek

- Teljes energia (mozgási és potenciális energiák összege):

$$E = \frac{m}{2} \sum_{i=1}^{N} \mathbf{v}_i^2 + \sum_{i \neq j} U(|\mathbf{r}_i - \mathbf{r}_j|)$$

- Hőkapacitás (a fluktuáció-disszipáció tétel alapján):

$$C_V = \left(\frac{\partial E}{\partial T} \right)_V = \frac{1}{k_B T^2} \left[\langle E^2 \rangle - \langle E \rangle^2 \right]$$

Az átlagolás elvileg sokaságátlag (pl. sok különböző véletlen futtatás eredménye), de egyensúlyban ezt jól közelíti az időátlag. A hőkapacitást a hőmérséklet-fluktuációkból is becsülhetjük:

$$\langle T^2 \rangle - \langle T \rangle^2 = \frac{3}{2} N (k_B T)^2 \left[1 - \frac{3 N k_B}{2 C_V} \right]$$

- Véges zárt rendszernél a nyomást mérhetnénk a doboz falánál történt impulzusváltozásokból is, de a Viriál-tétel alapján a következő kifejezés is használható:

$$PV = Nk_BT + \frac{1}{3}\left\langle \sum_{i<j} \mathbf{r}_{ij} \cdot \mathbf{F}_{ij} \right\rangle$$

A Kompresszibilitási faktor kifejezi, mennyire vagyunk távol az ideális gáz állapottól:

$$Z = \frac{PV}{Nk_BT} = 1 - \frac{1}{3Nk_BT}\left\langle \sum_{i<j} \mathbf{r}_{ij} \cdot \mathbf{F}_{ij} \right\rangle$$

Nagy sűrűségen a taszító potenciál miatt $Z > 1$, kisebb sűrűségeken pedig a vonzó van der Waals erők miatt $Z < 1$. Szabad részecskékre, ideális gázra lenne $Z = 1$.

Feladatok

1. Értsük meg az md.cpp programot és ábrázoljuk kimenetét! Értelmezzük a kapott görbét! Vizsgáljuk meg az md2.cpp programban a javításokat. A program futtatási eredményét elemezve vizsgáljuk meg, hogy mennyit segített a kezdeti feltételek pontosabb beállítása, és mennyi idő alatt áll be a kívánt hőmérséklet!
2. Értsük meg a md3.cpp program hogyan kezeli a levágásokat és a szomszédsági listát, és annak frissítését. Kísérletezzünk más r_{cutoff}, r_{max} és `updateInterval` paraméterekkel, nézzük meg hogyan változik a sebesség és a pontosság az eredeti md2.cpp programhoz képest.
3. Írjuk meg a teljes energiát, nyomást, hőkapacitást, kompresszibilitást számoló függvényeket! Próbáljuk meg a fázisátalakulási hőmérsékletet meghatározni a rendparaméterek (hőkapacitás, kompresszibilitás) figyelésével. Vessük össze az értékeket a valódi argon olvadás és forráspontjával! Hogyan befolyásolja N értéke a tapasztaltakat?

Szorgalmi feladatok: Lapozz!

Szorgalmi feladatok:

4. Alakítsuk át a programot úgy, hogy a periodikus határfeltétel helyett zárt, kemény fala legyen, ahonnan visszapattannak a részecskék! (Egyszerű megoldás a sebességek és pozíciók tükrözése, kicsit bonyolultabb a falat is tömör Lennard-Jones potenciállal leírható anyagnak venni.) Mérjük a nyomást a falakra kifejtett erő segítségével, és ellenőrizzük mennyire teljesül a $pV = Nk_BT$ törvény!
5. Értsük meg a md3-gl.cpp programban a 3D grafika használatát! Szerkesszük össze a kepler.cpp programmal úgy, hogy a gravitációs 3-test (vagy soktest) probléma dinamikáját ábrázolja. Vajon alkalmazható itt is egy levágási hossz a kód gyorsítására?