FYS4150 - Computational Physics

Molekylærdynamikk

Prosjekt 5

Thomas Haaland
homhaa@fys.uio.no
Kandidatnummer: 58

Øyvind Are Rysstad Johnsen
oajohnse@fys.uio.no
Kandidatnummer: 62

Sammendrag

I dette prosjektet benytter vi den deterministiske metoden *Molekylærdynamikk* til å simulere argon-atomer i en krystall, for ulike temperaturer. Vi modellerer interaksjonene mellom atomene - med periodiske grensebetingelser - og undersøker diverse statistiske egenskaper ved systemet.

Github-adresse:
https://github.com/thomashaaland/molecular-dynamics-fys3150

11.12.2015

Innhold

1 Introduksjon

I dette prosjektet tar vi utgangspunkt i kode for simulering av 100 ikke-vekselvirkende argon-atomer, uniformt fordelt i et område på $5 \times 5 \times 5$ enhetsceller (*molecular-dynamics-fys3150*, 2015) [2]. Koden fungerer som et skjelett for metoden Molekylærdynamikk, og gir atomene initielle hastigheter med Maxwell-Boltzmann-distribusjon. Vi implementerer periodiske grensebetingelser (som anvist på Wikipedia [5]), integratoren *Velocity Verlet* og *Lennard-Jones*-potensialet, setter den gjennomsnittelige bevegelsesmengden i systemet lik null og plasserer atomene i et flatesentrert, kubisk gitter. Fra dette systemet beregner vi de fysiske størrelsene temperatur, kinetisk energi, potensiell energi, total energi og diffusjonskonstanten. Vi finner også smeltetemperatur for krystallen, og variansen til den totale energien.

Wikipedia [5] forteller at Molekylærdynamikk blir brukt blant annet i molekylærbiologi, biokjemi, biofysikk og materialvitenskap. Molekylærdynamikk blir brukt for å simulere ting som ikke er direkte observerbart, som tynnfilm-vekst. Det blir også brukt for å undersøke fysiske egenskaper ved nanoteknologi som ennå ikke er blitt skapt. Metoden brukes også til å forutse egenskapene til proteiner og DNA/RNA i biologi.

2 Teori

2.1 Molekylærdynamikk

Wikipedia [5] forteller at Molekylærdynamikk er en deterministisk metode for datasimulering av mangelegemeproblemer med atomer og/eller molekyler. Ifølge encyklopedi-artikkelen benytter metoden numerisk løsning av Newtons bevegelseslover for mange interagerende parikler, der kreftene mellom partiklene er gitt fra forhåndsdefinerte potensialer eller kraftfelt - for å finne bevegelsesbanene til partiklene over et gitt tidsrom. Tanken bak denne metoden er å sample mikrotilstander fra et statistisk ensemble. I vår simulering har vi et mikrokaninisk ensemble med et konstant antall atomer, en konstant energi og et konstant volum. Alogoritmen for implementering av metoden kan beskrives ved fire steg, når vi benytter *Verlocity Verlet*-integrasjon som forklart i *Project 5, Molecular dynamics: atomic modeling of argon* (2015) [3]:

Algoritme:

1. Først, gi atomene startposisjoner og utgangshastigheter. Beregn kreftene mellom atomene.
2. Beregn hastigheten for et halv tidssteg, og benytt denne hastigheten til å finne den nye posisjonen for hele tidssteget. Ta i bruk randbetingelsene. Finn kreftene gitt disse endringene, og beregn hastigheten i slutten av tidssteget med de oppdaterte kreftene.
3. Beregn og skriv ut de fysiske størrelsene vi ønsker å finne.
4. Start et nytt tidssteg, og gjenta fra og med punkt to så mange ganger som ønsket.

Project 5, Molecular dynamics: atomic modeling of argon (2015) [3] forklarer oss at Molekylærdynamikk er en metode som lar oss undersøke faserommet til atomene, beskrevet ved sannsynlighetene vi kan beregne med statistisk mekanikk - og ikke gir oss de sanne bevegesesbanene for partiklene. Modellen bygger på antakelsen om at alle tilgjengelige mikrotilstander vil være like sannsynlige for systemet, over lengre tid. Noe som begrenser metoden er at mens den er god til å simulere med klassisk mekanikk som utgangspunkt blir beregningene med kvantemekanikk fort så tunge at metoden er mindre nyttig i praksis.

2.2 Lennard-Jones-potensialet

Rabia Naeem [4] forteller at Lennard-Jones-potensialet beskriver den potensielle energien i interaksjonene mellom to ikke-bindende atomer (eller molekyler) som en funksjon av avstanden mellom dem - og tar

hensyn både til tiltrekkende og frastøtende krefter. For Argon vil dette potensialet gi god overstemmelse med eksperimentelle verdier for ulike termodynamiske egenskaper - tillegg til å få fram faseoverganger. Tilstandslikningen for partikkel-systemet er van der Waals-likningen. Potensialet er gitt som:

$$U(r_{ij}) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \tag{1}$$

Her er σ avstanden mellom atomene der de intermolekylære kreftene er lik null og r_{ij} avstanden mellom atomene. Vi ser at det første leddet vil dominere for små r_{ij} - og virke frastøtende, mens det andre leddet styrer potensialet for store - og virker tiltrekkende. For å forklare hvilken betydning ϵ har, kan vi først derivere uttrykket for potensialet:

$$\begin{aligned} \frac{dU}{dr_{ij}} &= 4\epsilon \left[-\frac{12}{\sigma} \times \left(\frac{\sigma}{r_{ij}} \right)^{13} + \frac{6}{\sigma} \times \left(\frac{\sigma}{r_{ij}} \right)^{7} \right] \\ &= 24\frac{\epsilon}{\sigma} \left[\left(\frac{\sigma}{r_{ij}} \right)^{7} - 2\left(\frac{\sigma}{r_{ij}} \right)^{13} \right] \end{aligned} \tag{2}$$

I bunnpunktet vil den deriverte være lik null. Fra dette kan vi finne avstanden r_{bunn} mellom atomene:

$$\begin{aligned} \frac{dU}{dr_{ij}}(r_{bunn}) = 0 &= 24\frac{\epsilon}{\sigma} \left[\left(\frac{\sigma}{r_{bunn}} \right)^{7} - 2\left(\frac{\sigma}{r_{bunn}} \right)^{13} \right] \\ &= \left(\frac{\sigma}{r_{bunn}} \right)^{7} \times \left[1 - 2\left(\frac{\sigma}{r_{bunn}} \right)^{6} \right] \\ &= 1 - 2\left(\frac{\sigma}{r_{bunn}} \right)^{6} \\ r_{bunn} &= 2^{\frac{1}{6}}\sigma \end{aligned} \tag{3}$$

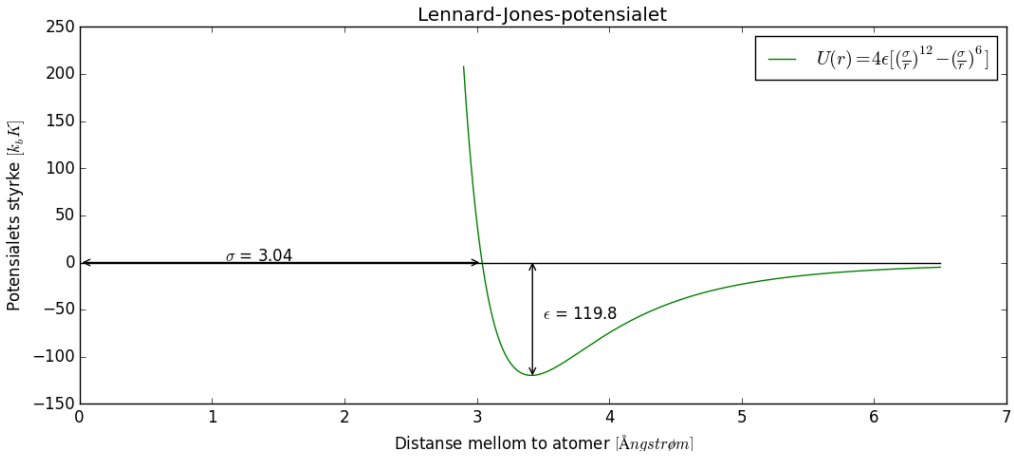
Vi setter inn denne avstanden i likning (1) for å finne potensialet:

$$\begin{aligned} U(r_{bunn}) &= 4\epsilon \left[\left(\frac{\sigma}{r_{bunn}} \right)^{12} - \left(\frac{\sigma}{r_{bunn}} \right)^{6} \right] \\ &= 4\epsilon \left[\left(\frac{1}{2^{\frac{1}{6}}} \right)^{12} - \left(\frac{1}{2^{\frac{1}{6}}} \right)^{6} \right] \\ &= 4\epsilon \left[\frac{1}{4} - \frac{1}{2} \right] \\ &= -\epsilon \end{aligned} \tag{4}$$

I figur (1) viser vi Lennard-Jones-potensialet for $\frac{\epsilon}{k_b} = 119.8\text{K}$ (der k_b er Boltzmanns konstant) og $\sigma = 3.04\text{Å}$. Vi ser at ϵ tilsvarer dybden i potensialbrønnen, og fungerer som et mål på styrken i potensialet - jo større ϵ, jo sterkere interaksjoner mellom atomene. Vi finner kreftene som virker mellom atomene ved å ta den negative gradienten av potensialet:

$$\mathbf{F}(r_{ij}) = -\nabla U(r_{ij}) \tag{5}$$

Komponentvis har vi da:



Figur 1: Figuren viser Lennard-Jones-potensialet, med verdiene for σ og ϵ som vi benytter i prosjektet. Kurven har en tydelig dupp.

$$
\begin{aligned}
F_x &= -\frac{\partial U}{\partial r_{ij}} \frac{\partial r_{ij}}{\partial x_{ij}} \\
&= 24\epsilon \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \times \frac{x_{ij}}{r_{ij}^2}
\end{aligned} \tag{6}
$$

Og tilsvarende for y- og z-komponentene.

2.3 Velocity Verlet

I Molekylærdynamikk benyttes som oftest en *Verlocity Verlet*-integrator. Denne integratoren er symplektisk, og fungerer derfor godt til å bevare geometrien i partikkelbevegelsene. Energien driver heller ikke over tid ved bruk av Velecity Verlet, som den kan gjøre for andre algoritmer. Integratoren kan beskrives ved tre steg, om vi ser bort ifra beregning av kreftene:

$$\mathbf{v}(t + \Delta t/2) = \mathbf{v}(t) + \frac{\mathbf{F}(t)}{m} \frac{\Delta t}{2} \tag{7}$$

$$\mathbf{r}(t + \Delta t) = \mathbf{r} + \mathbf{v}(t + \Delta t/2) \Delta t \tag{8}$$

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t + \Delta t/2) + \frac{\mathbf{F}(t + \Delta t)}{m} \frac{\Delta t}{2} \tag{9}$$

2.4 Fysiske egenskaper

Vi kan finne ulike fysiske egenskaper ved et ensemble av atomer - som kinetisk energi, potensiell energi, temperatur og diffusjonskonstant. Den totale kinetiske energien får vi ved å summere over den kinetiske energien til hvert atom:

$$E_k = \sum_i \frac{1}{2} m_i v_i^2 \tag{10}$$

Den totale potensielle energien i et system av atomer finner vi ved å summere over alle atomparene:

$$V = \sum_{i>j} U(r_{ij}) \tag{11}$$

For Lennard-Jones-potensialet er U som i likning (1). Temperaturen kan vi få et uttrykk for fra ekvipartisjonsprinsippet:

$$\langle E_k \rangle = \frac{3}{2} N_{atoms} k_B T \tag{12}$$

Momentantemperatur kan vi da uttrykke ved:

$$T = \frac{2}{3} \frac{E_k}{N_{atoms} k_B} \tag{13}$$

Diffusjonskonstanten, D, finner vi fra Einsteins relasjon:

$$\left\langle r^2(t) \right\rangle = 6Dt \tag{14}$$

$$D = \frac{\left\langle r^2(t) \right\rangle}{6t} \tag{15}$$

Her er $\left\langle r^2(t) \right\rangle$ den gjennomsnittelige kvadratet av avstanden atomene har beveget seg fra utgangsposisjonen i systemet. For atom i er denne avstanden:

$$|\mathbf{r}_i(t) - \mathbf{r}_i(0)|^2 \tag{16}$$

Vi ser av uttrykket for diffusjonskonstanten at i en fast krystallstruktur vil den synke etter som tida går - siden utslagene fra atomenes initielle posisjoner vil være små og ikke økende med tid. Når temperaturen øker, slik at krystallstrukturen brytes - og det forekommer en faseovergang, forteller uttrykket oss at diffusjonskonstanten vil kunne vokse mens atomene blir friere til å vandre - helt til faseovergangen er over. Da vil D stabilisere seg (for en gitt temperatur) siden vandringen atomene atomene kan gjøre i et gitt tidintervall ikke lenger endrer seg med tid.

3 Metode og kodeimplementering

3.1 Kodestruktur

Vi bruker programstrukturen som gitt til oss av Hafreager (2015) [2]. Vi lagde flowchart til koden som vedlegg. Vi tok utgangspunkt i denne koden og lagde vår egen kode der det var behov som ny integrator, implementerte potensialet og randpunktbetingelser. Koden benytter seg av klasser og moduler. I koden lages objektet *system* som vi jobber opp imot. Objektet *system* er en instans av klassen *System*. Dette objektet er sentralt og behandles av moduler gjennomgående i koden. Koden er dermed objektorientert.

3.2 Grense- og start- betingelser

Når vi implimenterer koden starter vi med å sette på grensebetingelser. Vi gjør dette ved å danne en boks rundt systemet vårt som er slik at dersom et atom kommer utenfor boksen, flyttes den over til den andre siden av boksen som vist i koden under.

```
void System::applyPeriodicBoundaryConditions()
    if (position(i) > 0.5 * m_systemSize[i]) {position(i) -= m_systemSize[i];}
    if (position(i) < -0.5 * m_systemSize[i]) {position(i) += m_systemSize[i];}
```

Vi gjør det samme med kreftene, og det viser seg at vi kan bruke akkurat samme kode, men sende inn forskjellen i posisjon $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ i stedenfor posisjon siden potensialet er som i likning (1).

Fordelen med slike grensebetingelser er at systemet nå oppfører seg tilnærmet som om det var uendelig stort, siden hvert atom føler at de er i midten av en slik boks, men krefter fra atomer lenger borte, vil ikke føles. Siden kreftene fra atomer så langt borte vil være små, burde dette være en god tilnærmelse

til et uendelig stort system. Fordelen med et uendelig stort system er at vi ikke trenger å bry oss med grensene der systemet slutter, og systemet vil være enklere å simulere, siden vi kan klare oss med relativt få partikler og allikevel få en god tilnærming på et stort system.

Videre implementerer vi startbetingelser. Vi vil at systemet skal starte i en stabil struktur, slik at vi har bedre kontroll på når systemet smelter. Vi gjør dette ved å dele opp systemet vårt i $5 \times 5 \times 5$ kuber, der hver kube blir fyllt med fire atomer slik at

$$\mathbf{r}_1 = 0\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + 0\hat{\mathbf{k}}$$

$$\mathbf{r}_2 = \frac{b}{2}\hat{\mathbf{i}} + \frac{b}{2}\hat{\mathbf{j}} + 0\hat{\mathbf{k}}$$

$$\mathbf{r}_3 = 0\hat{\mathbf{i}} + \frac{b}{2}\hat{\mathbf{j}} + \frac{b}{2}\hat{\mathbf{k}}$$

$$\mathbf{r}_4 = \frac{b}{2}\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + \frac{b}{2}\hat{\mathbf{k}}$$

der b er lengden på en en side av en boks. Koden for å få til dette ble

```
void System::createFCCLattise()
    double x[4] = {0, 0.5 * b, 0,       0.5 * b};
    double y[4] = {0, 0.5 * b, 0.5 * b, 0};
    double z[4] = {0, 0,       0.5 * b, 0.5 * b};

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            for (int k = 0; k < N; k++) {
                for (int l = 0; l < 4; l++) {
                    Atom *atom = new Atom(UnitConverter::massFromSI(6.63352088e-26));
                    atom->position.set(b * i + x[l],b * j + y[l],b * k + z[l]);
                    atom->initialPosition = atom->position;
                    atom->resetVelocityMaxwellian(temperature);
                    m_atoms.push_back(atom);
                }
            }
        }
```

3.3 Integratorimplementering

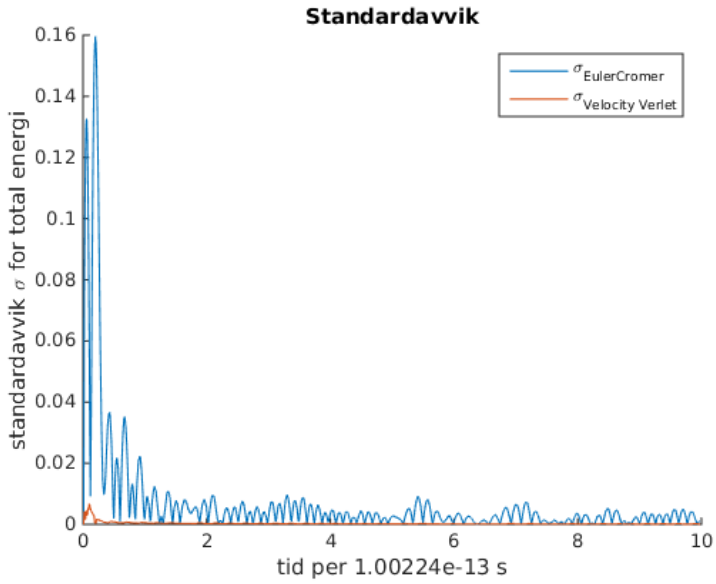
Vi fikk utdelt Euler Cromer integratoren som en begynnelse. Vi implementerer krefter ved hjelp av likning (5) og skriver koden

```
void LennardJones::calculateForces()
    for (i = 0 to N) {
        for (j = i to N) {
            fIJ = prefix * 6 * (2 * uIJFirstArgument - uIJSecondArgument)/rIJSquare*rIJ;
            totalFi[i] += fIJ;
            totalFi[j] -= fIJ;
        end
    end
```

mens vi passer på grensebetingelser på måten nevnt i forrige seksjon.

Som et alternativ til Euler Cromer implementerte vi Velocity Verlet (likning (9)). For å gjøre dette brukte vi koden

```
void VelocityVerlet::integrate()
    atom->velocity += 0.5 * atom->force*dt / atom->mass();
    atom->position += atom->velocity*dt;
    system->applyPeriodicBoundaryConditions(atom->position, atom->initialPosition);
```



Figur 2: Standardavvik for Euler Cromer og Velocity Verlet. Velocity Verlet sitt avvik er tydelig mindre enn Euler Cromer

Vi passer på å beregne kreftene en gang før loopen starter slik at vi ikke mister en iterasjon.

3.4 Datainnsamling

Når vi nå har en fungerende simulering kan vi begynne å samle inn data. Siden hastigheten i hvert tidssteg er kjent kan vi kjapt finne kinetisk energi fra likning (10). Fra potensialet vil vi finne potensiell energi gitt av likning (11). Summen av disse er totale energien. Videre finner vi momentane temperatur fra ligning (13)

Vi skal også finne diffusjonskonstanten. Vi bruker likning (15) som vi implimenterer med koden

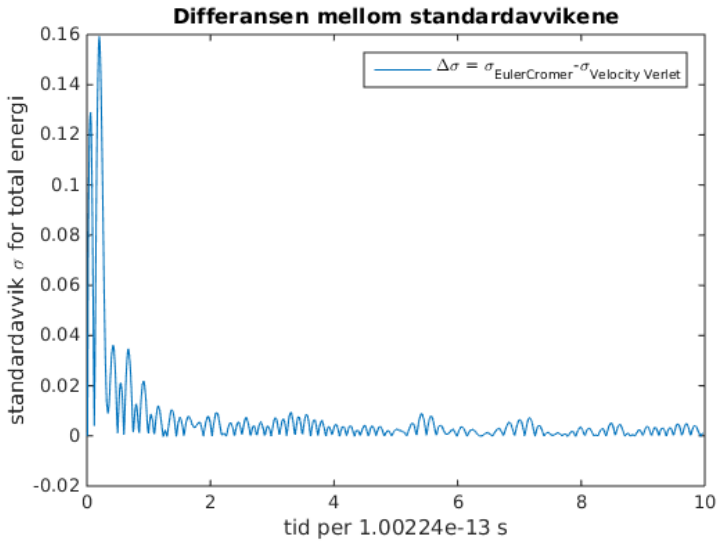
```
void StatisticsSampler::sampleDiffusionConstant()
    m_diffusionConstant = 0;
    for (Atom *atom : system.atoms()) {
        m_diffusionConstant += (atom->position-atom->initialPosition).lengthSquared();
    }
    m_diffusionConstant /= (6 * system.time() * system.atoms().size());
```

Her måtte vi endre funskjonen for grensebetingelser også slik at dersom et atom blir flyttet på grunn av at det når kanten, må vi flytte initial posisjon også. Dette er viktig siden dersom vi ikke gjør dette vil r_{ij} vokse siden r_i blir endret, men ikke r_j.

4 Resultater og diskusjon

4.1 Integratorer

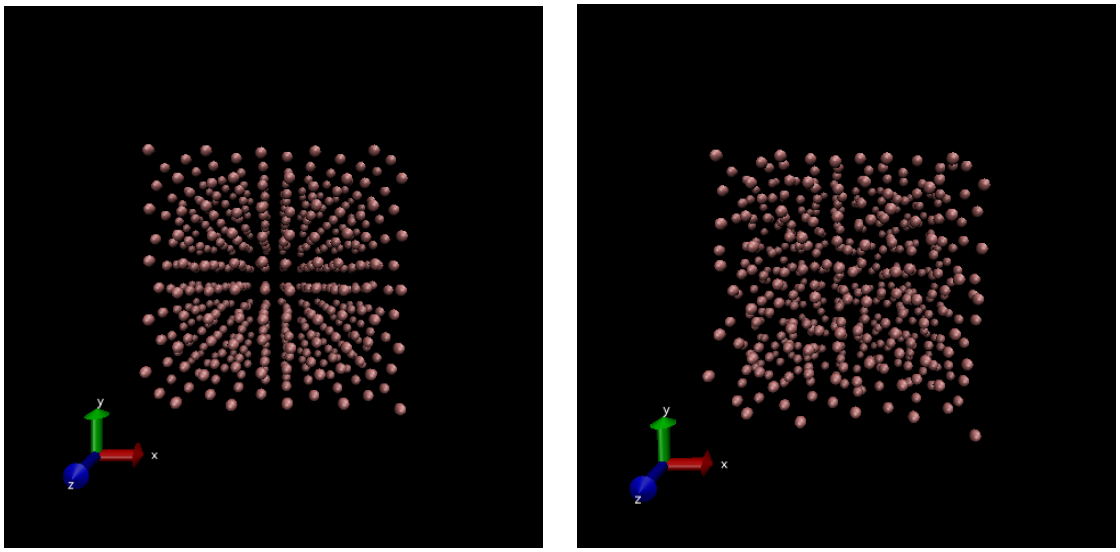
Vi sammenligner de to integratorene ved å skrive ut standardavviket σ av totalenergien over tid.
Fra figurene (2) og (3) har Euler Cromer stadig en ganske stor feil. Dette kan vi anta vil raskt føre til voksende feil i energi. Euler Cromer er mer sårbar for høyere temperturer også siden potensialet vi jobber med er som vi ser av figur (1) en bratt kurve når to atomer er nær hverandre. Da vil vi få et ekstra bidrag til energi, siden atomene vil nærme seg hverandres sentrum raskere enn integratoren vil



Figur 3: Her er avviket til Velocity Verlet trekkes fra Euler Cromer.

kunne kompansere for. Fra dette ser vi som forventet at Velocity Verlet er mer stabil og dermed mer hensiktsmessig for Molekylærdynamikk.

4.2 Visualisering



(a) Krystallstruktur

(b) Smeltet krystall

Figur 4: Skjermbilde av VMD, for henholdsvis den initelle krystallstrukturen og etter at materialet har smeltet.

Med hjelp av VMD (Visual Molecular Dynamics) får vi visualisert oppførselen til atomene. Da ser vi at systemet er stabilt for lave nok temperaturer og atomene holder seg i nærheten av sine opprinnelige plasser. Med øyemål mener vi at krystallen begynner å smelte ved en initiell temperatur på $60000K$ og er tydelig smeltet ved $90000K$. et er vanskelig å se med blotte øye, men vi føler oss sikre på at argonet

tidssteg	totalenergi
1	-2744.18
101	-2741.98
201	-2742.21
301	-2742.27

Tabell 1: Total Energi [eV] for $T = 40000K$

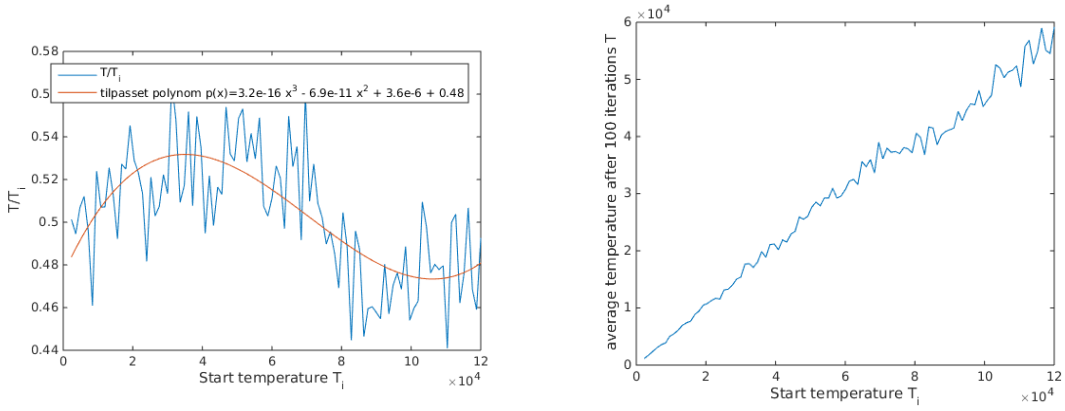
er smeltet over $80,000K$ siden atomene vandrer tydelig rundt, mens de er frosset under $60,000K$ siden atomene tydelig holder gitterstrukturen.

Siden vi har plassert $4 \cdot 5 \times 5 \times 5 = 500$ atomer i en boks som har et volum på $5b \times 5b \times 5b = 125b^3 = 125 \cdot (5.26)^3 \text{Å}^3 = 1.819 \times 10^{-26} m^3$ er tettheten $\rho = \frac{N}{V} = \frac{500}{1.819 \times 10^{-26} m^3} = 2.749 \times 10^{10}$ atomer per $\mu m^3 = 1824 \frac{g}{L}$ som er litt tettere enn vann.

4.3 Datainnsamlinger

Egenskapene vi fører statistikk på er momentantemperatur, kinetisk-, potensiell-, totalenergi og diffusjonskonstanten. Temperaturen er basert på kinetisk energi som forklart i likning (13) og dermed det er hovedsakelig temperatur og diffusjonskonstanten vi har valgt å drøfte. Vi ser at totalenergi holder seg nært konstant. Foreksempel er totalenergi for temperatur $T_i = 40000K$ i tabell (1) nær konstant over 300 tidssteg. At den holder seg konstant er en indikator på at integratoren vår ikke gir så store feil at systemet oppfører seg ufysisk og ble diskutert nærmere i delavsnittet om integrator resultater.

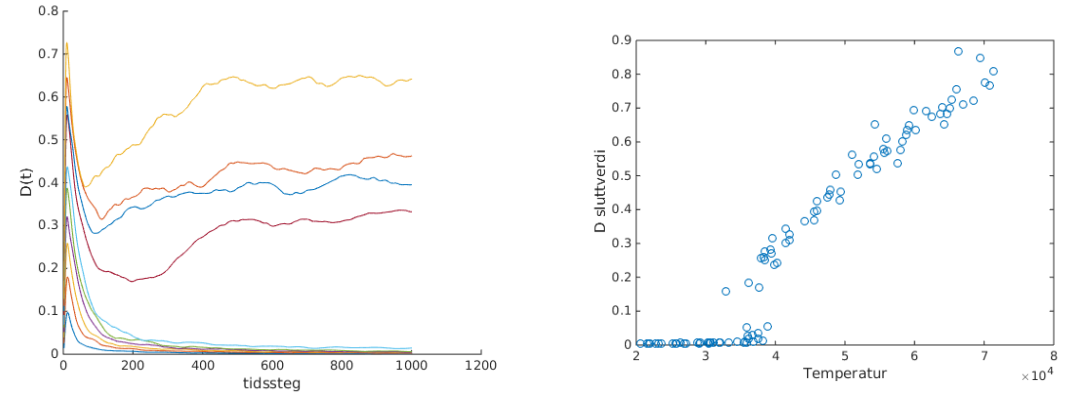
Når vi behandler temperatur ser vi fra figur (5a) at forholdstallet mellom momentantemperatur og initiell temperatur øker fram til systemet begynner å smelte. Dette forholdstallet er mindre enn 1. Vi ser at dette kommer av at atomene i utganspunktet er plassert i potential minimumene. Dermed vil alltid noe kinetisk energi konverteres til potensiell energi, og momentantemperaturen vil dermed synke. Selv om det er en del støy i systemet er denne tendensen tydelig. Hadde vi hatt flere atomer vil vi tro at støyen ville blitt mindre, men vi prøvde kun med 500 atomer. Fra visualiseringen så vi at systemet begynte å smelte ved $60000K$ og var smeltet ved $90000K$. Fra figur (5a) virker det som at denne oppførselen gjentar seg. Når krystallen smelter konverteres mer kinetiskenergi til potensiellenergi siden atomene nå har nok energi til å bryte ut av potensialbrønnene. Fra figur (5b) virker det som at temperaturendringen er nær null mens krystallen smelter. Til tross for støy synes vi å kunne se en slik tendens. Fra denne grafen later smeltingen til å skje mellom $70000K$ og $90000K$.



(a) (b)

Figur 5: Til venstre har vi forholdet mellom momentantemperatur og initielle temperaturer som en funksjon av initialtemperatur. Legg merke til at forholdet er mindre enn 1. Vi har også tegnet på et tilpasset polynom for å fremheve oppførselen til forholdstallet. Til høyre ser vi på y-aksen gjennomsnittstemperatur samplet etter 100 tidssteg, plottet mot initialtemperatur.

Vi ser også av figur (5b) og figur (5a) at dersom vi ønsker en momentantemperatur gir disse grafene hvilken initiell temperatur vi må ha for å få denne.



(a) (b)

Figur 6: Til venstre har vi diffusjonskonstanter over tid. Konstantene for temperaturer under smeltepunktet samler seg mot null. Konstantene for temperatur over smeltepunktet konverger rundt 0.3 og 0.6. Til høyre har vi valgt ut konstanten fra det siste tidssteget og plottet verdien som en funksjon av temperatur. Her bruker vi simulering for langt flere temperaturer, for å få et tydeligere bilde av hvordan faseovergangen ser ut.

Når vi vurderer diffusjonskonstanten ser vi at før systemet er smeltet er konstanten nær null men den vokser ganske raskt til ca. 0.3 eller mer når systemet har smeltet. Fra figur (6b) virker det som at konstanten gjør et byks ved $T = 37000K$ og vi tolker dette som at atomene ikke lenger holder en krystallstruktur men er frie til å bevege seg mellom potensialbrønner. Dette er som vi ville ha forventet fra teorien. Fra figur (6a) ser vi at diffusjonskonstanten vokser umiddelbart etter at simulering har begynt, før den begynner å stabilisere seg etter omtrent 600 tidssteg. Vi tolker dette som at det kommer av at alle atomene starter i minimum av potensialet med maksimal kinetisk energi, så i den første perioden vil alle atomer bevege seg bort fra startposisjon. Etter et par perioder vil atomene ikke lenger bevege

seg over potensial minima i samme periode og vi får et gjennomsnitt av forflytning. Fra figur (6a) virker det som at dette skjer etter 600 tidssteg.

5 Konklusjon

I dette prosjektet har vi sett at Molekylærdynamikk er nyttig for å finne og forstå statistiske egenskaper ved et system av atomer. Blant annet kan metoden brukes til å vise faseoverganger og undersøke hvordan ulike fysiske størrelser endrer seg for ulike temperaturer. Vi har sett at integratoren Velocity Verlet fungerer tilfredstillende for denne metoden så lenge temperaturen ikke blir så høy at tidssteget må minskes. For vårt formål ser Lennard-Jones-potensialet ut til å fungere godt. Vi kunne gjort en mer naturnær simulering ved å modellere elektronsky- og proton-vekselvirkninger. Men med vår metode blir slike hensyn tatt med implisitt i potensialet. Vi får dermed modellert van der Waals-krefter og skjerming med en enkel programeringskode. Med mer kompliserte potensialer, for eksempel et som har verdier avhengig av retning, ser vi hvordan man kunne modellert for eksempel vann og proteiner.

I simuleringen vår har Argon en smeltetemperatur på like under $T_i = 40000K$. Dette er langt mer enn Argon ved en atmosfæres trykk som ligger på $83.8K$ (Wikiepdia [1]). Dette kan forklares med at vår Argon har en tetthet på $1824\frac{g}{L}$. Til sammenligning har Argon ved $0\ \deg C$ og $101,325\ kPa$ er tettheten $1,784\frac{g}{L}$. Dette viser at trykket vi simulerer ved er stort og vi ville forvente en høy smeltetemperatur.

Referanser

[1] Argon. (i. d.). I Wikipedia. Hentet 30. november, 2015, fra:
en.wikipedia.org/wiki/Argon

[2] Hafreager, A. (November, 2015). molecular-dynamics-fys3150 [Dataprogram]. Hentet 25. november, 2015, fra:
github.com/andeplane/molecular-dynamics-fys3150

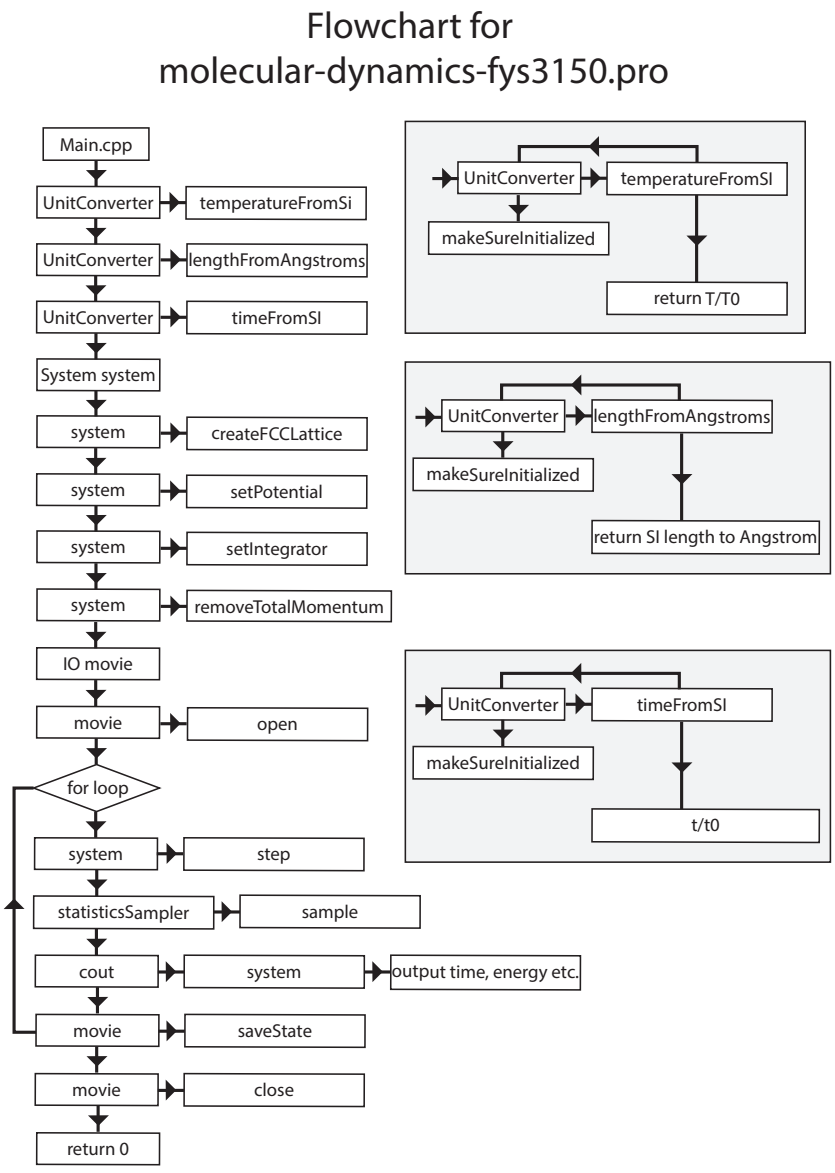
[3] Hjort-Jensen, M. (November, 2015). Project 5, Molecular dynamics: atomic modeling of argon. Hentet 6. desember, 2015, fra:
github.com/CompPhysics/ComputationalPhysics1/blob/gh-pages/doc/Projects/Project5/project5_md.pdf

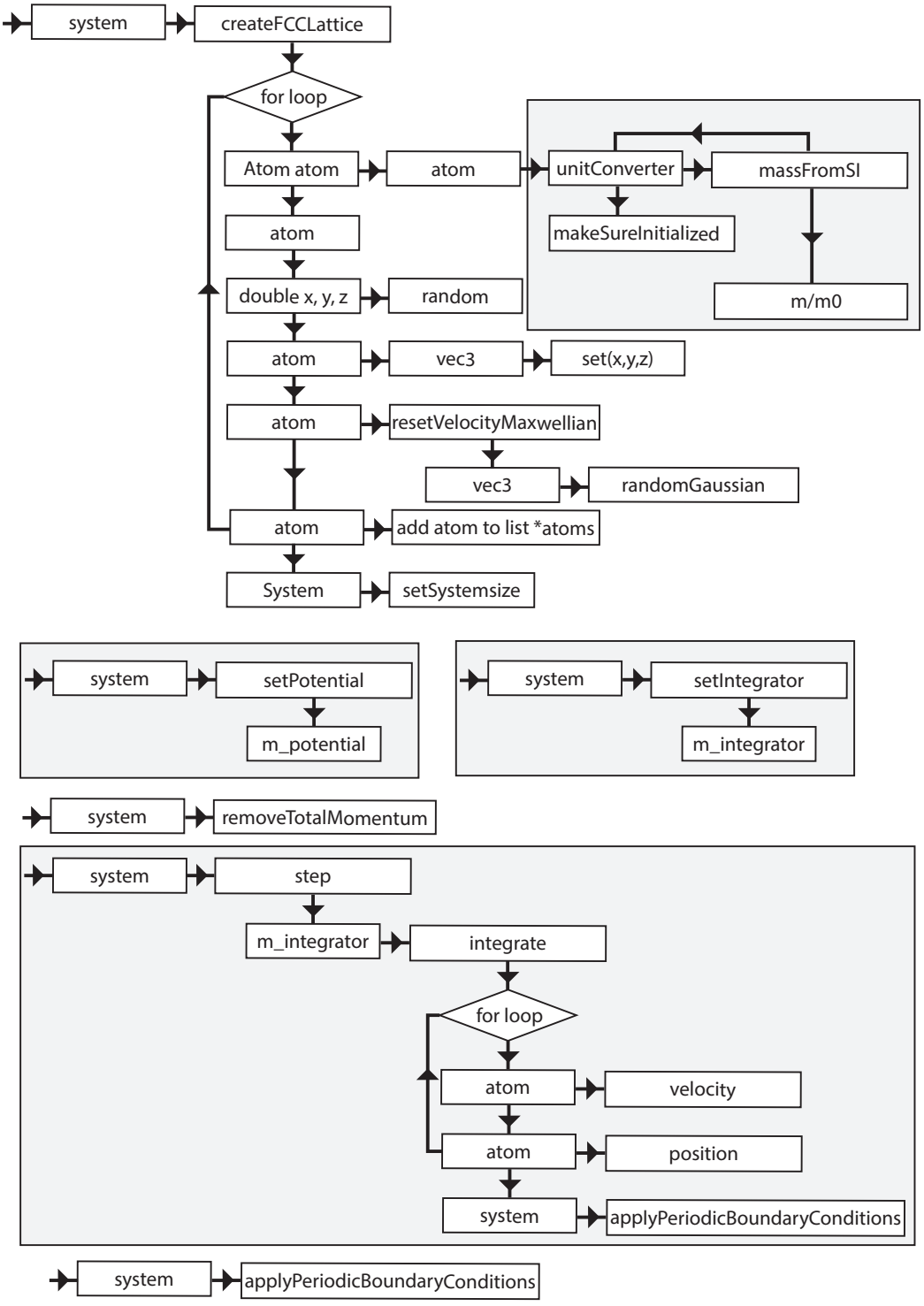
[4] Lennard-Jones Potential. (i. d.). Hentet 9. desember, 2015, fra:
chemwiki.ucdavis.edu/Physical_Chemistry/Physical_Properties_of_Matter/...
...Intermolecular_Forces/Lennard-Jones_Potential

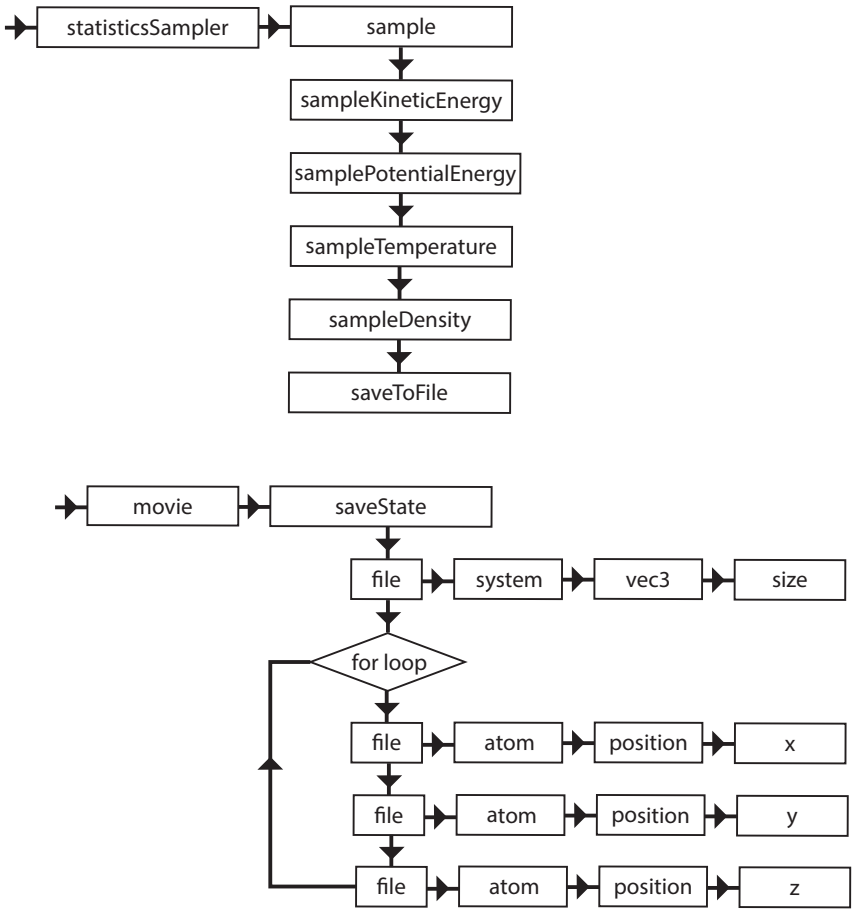
[5] Molecular Dynamics. (i. d.). I Wikipedia. Hentet 30. november, 2015, fra:
en.wikipedia.org/wiki/Molecular_dynamics

6 Vedlegg

6.1 Flowchart for utdelt kode




system
createFCCLattice
for loop
Atom atom
atom
unitConverter
massFromSI
makeSureInitialized
m/m0
atom
double x, y, z
random
atom
vec3
set(x,y,z)
atom
resetVelocityMaxwellian
vec3
randomGaussian
atom
add atom to list *atoms
System
setSystemsize
system
setPotential
m_potential
system
setIntegrator
m_integrator
system
removeTotalMomentum
system
step
m_integrator
integrate
for loop
atom
velocity
atom
position
system
applyPeriodicBoundaryConditions
system
applyPeriodicBoundaryConditions


statisticsSampler
sample
sampleKineticEnergy
samplePotentialEnergy
sampleTemperature
sampleDensity
saveToFile
movie
saveState
file
system
vec3
size
for loop
file
atom
position
x
file
atom
position
y
file
atom
position
z