Computergestütztes wissenschaftliches Rechnen SoSe 2004

Alexander K. Hartmann, Universität Göttingen

10. Juni 2004

6.4 Temperatur

Gleichverteilungssatz: im Mittel $\frac{1}{2}k_BT$ kinetische Energie pro Freitheitsgrad, d.h.

$$\sum_i \frac{1}{2} m_i \langle \dot{\underline{r}}_i^2 \rangle = \frac{3N}{2} k_B T \tag{1}$$

Problem: Integration der Bewegungsgleichungen lässt Gesamtenergie konstant. Einfachste Lösung: Reskalierung der Geschwindigkeiten, so dass (1) erfüllt ist, d.h. man multipliziert jede Geschwindigkeit mit

$$\lambda = \sqrt{\frac{3k_BT}{\sum_i m_i \dot{\underline{r}}_i^2}} \tag{2}$$

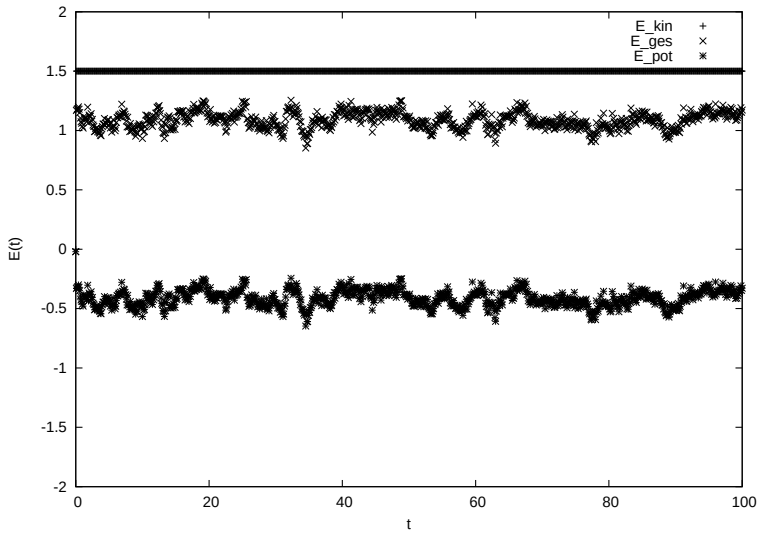


Abbildung 1: Kinetische, Gesamt- und potentielle Energie eines LJ Systems mit Reskalierung auf $k_BT = 1$ ($L = 10$, 50 Teilchen, $m = 1$, $\epsilon = 1$, $\sigma = 1$, $\tau = 0.001$).

Nachteil: Kinetische Energie = konstant. → Nosé-Hoover Thermostat. Grundidee: Einführung eines "Temperatur-Bad" Teilchens, das mit allen Atomen wechselwirkt und falls $\sum_i m_i \underline{v}_i^2 < 3Nk_BT$ Temperatur langsam zuführt (bzw abführt).

6.5 Exkursion: Povray

Programm zur 3D Darstellung von beliebigen Szenen: `povray`.
Grundidee: Szene = Ansammlung von Objekten + Lichtquellen + Kamera. Wird in einem *Szenebeschreibungsfile* (`.pov`) File im ASCII Format beschrieben.

Beispiel: Drei Kugeln mit Stangen verbunden auf Ebene (`example.pov`)

```
#include "colors.inc"
background { color White }
sphere {  <10, 2, 0>, 2 pigment { Blue } }
cylinder { <10, 2, 0>,  <0, 2, 10>, 0.7 pigment { color Red } }
sphere {  <0, 2, 10>, 4 pigment { Green transmit 0.4} }
cylinder { <0, 2, 10>,  <-10, 2, 0>, 0.7 pigment { Red } }
sphere {  <-10, 2, 0>, 2 pigment { Blue } }
plane { <0, 1, 0>, -5 pigment { checker color White, color Black}}
light_source { <10, 30, -3> color White}
camera {location <0, 8, -20>
    look_at  <0, 2,  10>
    aperture 0.4}
```

Povray *rendert* die Szene, d.h. berechnet welche Wege das Licht, das die Kamera erreicht, mit Reflxion, Brechung und Absorption nimmt → photorealistisch.
Aufruf:
`x-povray +I example.pov -display 0:0 -W640 -H480 +P`
Bedeutung der Optionen:

- `+I <file>`: Name des Inputfiles
- `-display <d>`: Wo wird es angezeigt
- `-W <w>`: Größe in x-Richtung
- `-H <h>`: Größe in y-Richtung
- `+P`: warte auf Klick in Window mit Bild, bevor es gelöscht wird.

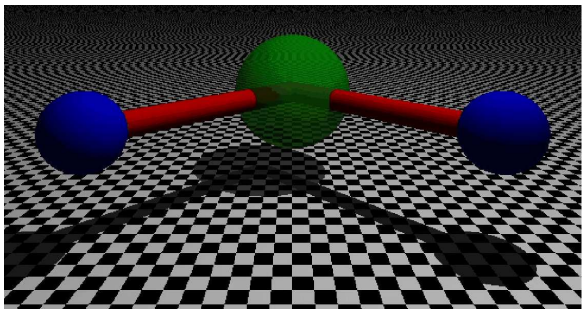


Abbildung 2: Beispielszene, die mit *Povray* gerendert wurde.

Hier: Unterroutine, die eine Atomkonfiguration in eine Povrayszene als Ansammlung von Kugeln umwandelt → 3d Visualisierung der Konfiguration.

```
/***************** glas_plot_cfg() ********************/
/** Prints configuration 'atom' to povray file with    **/
/** name cfg.pov. Display with x-povray +I cfg.pov     **/
/** PARAMETERS: (*)= return-paramter                   **/
/**     system: global system parameters               **/
/**       atom: atom data                              **/
/** RETURNS:                                           **/
/**      nothing                                       **/
/*********************************************************/
void glas_plot_cfg(glas_system_t *system, glas_atom_t *atom)
{
  char filename[1000];
  FILE *povfile;
  int t, d;                                    /* loop counters */
  double *r;                                        /* position */

  r = (double *) malloc(system->dim*sizeof(double));
  sprintf(filename, "cfg.pov");
  povfile = fopen(filename, "w");                  /* open file */
  fprintf(povfile, "#include \"colors.inc\"\n"   /* header etc */
          "#include \"shapes.inc\"\n\n");
  fprintf(povfile, "background { color Yellow }\n\n");
  fprintf(povfile, "camera {\n  location <%f, %f, %f>\n",
          0.5*system->l[0], -1.5*system->l[1], 0.5*system->l[2]);
  fprintf(povfile, "  sky <0,0,1>\n");
  fprintf(povfile, "  look_at  <%f, %f, %f>\n}\n\n",
          0.5*system->l[0], 0.5*system->l[1], 0.5*system->l[2]);
  fprintf(povfile, "  light_source { <%f, %f, %f> color White}\n\n",
          0.5*system->l[0], -0.5*system->l[1], 1.5*system->l[2]);
  fprintf(povfile, "  light_source { <%f, %f, %f> color White}\n\n",
          -0.5*system->l[0], -0.5*system->l[1], 1.5*system->l[2]);

  for(t=0; t<system->N; t++)                       /* print atoms */
  {
    for(d=0; d<system->dim; d++)
    {
      r[d] = atom[t].x[d];
      if(r[d] < 0)                    /* fold positions into box */
        r[d] += floor(-r[d]/system->l[d]+1)*system->l[d];
      if(r[d] > system->l[d])
        r[d] -= floor(r[d]/system->l[d])*system->l[d];
    }
    if(system->dim == 3)
      fprintf(povfile,
              "sphere { <%f,%f,%f>, %f\n    pigment { Blue }}\n",
              r[0], r[1], r[2], 0.4*atom[t].sigma);
  }
  fclose(povfile);
  free(r);
}
```

Startkonfiguration mit hexagonal dichtester Kugelpackung in Box der Größe $6\sigma_{\min} \times 6\frac{\sqrt{3}}{2}\sigma_{\min} \times 6\sqrt{\frac{8}{12}}\sigma_{\min}$, $N = 6x6x6$ Atome, 1000 Schritte.
Simulation bei hoher Temperatur $T = 10$ ergibt:

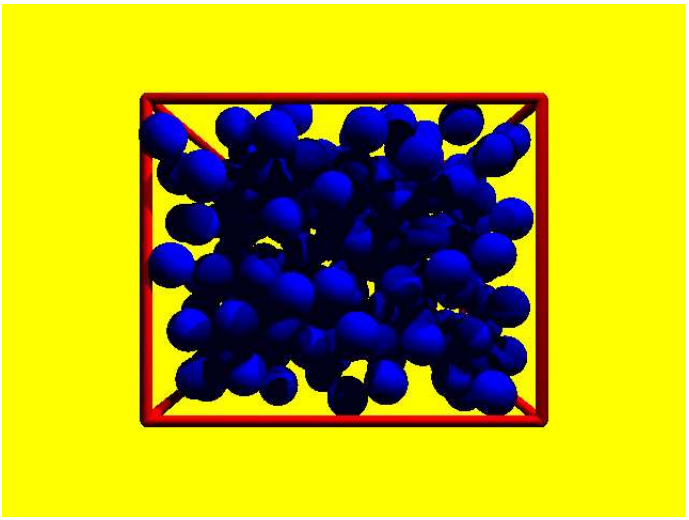


Abbildung 3: Konfiguration bei hoher Temperatur $T = 10$

Simulation bei niedriger Temperatur $T = 0.3$: System bleibt kristallin!

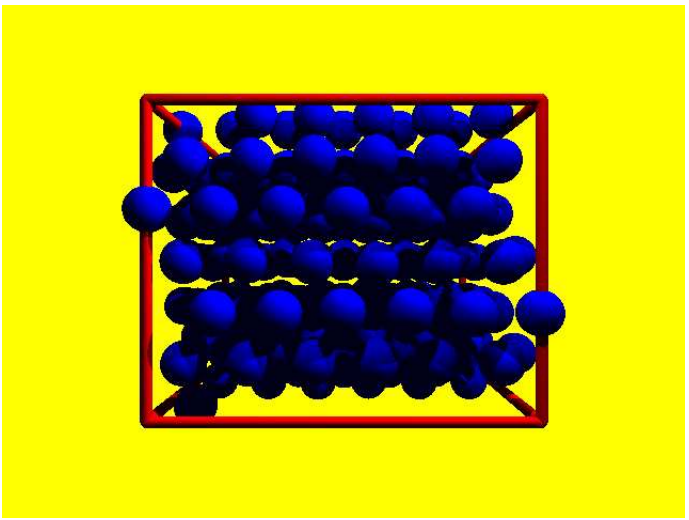


Abbildung 4: Konfiguration bei tiefer Temperatur $T = 0.3$

6.6 Reduzierte Einheiten

Für Lennard-Jones Systeme mit nur einer Teilchensorte: Man kann alle Größen in Einheiten der Teilchemasse m, des Radius σ und des Energieparameters ϵ ausdrücken → *reduzierte Einheiten*, z.B. reduzierter Abstand $r^\star = r\sigma$:

- Dichte $\rho^\star = \rho\sigma^3$
- Temperatur $T^\star = k_B T/\epsilon$
- Energie $E^* = E/\epsilon$

- Druck $P^* = P\sigma^3/\epsilon$
- Zeit $t^\star = t\sqrt{\epsilon/m\sigma^2}$
- Kraft $f^\star = f\sigma/\epsilon$
- ...

Vorteil: Man kann sich Berechungen im Programm sparen + alle Größen sind näher an $O(1)$.
Beispiel: Argon: $\epsilon = 121k_B = 1.67 \times 10^{-21}$J, $\sigma = 3.4$Å, $m = 6,63 \times 10^{-26}$

6.7 Berechnung des Drucks

Vorgriff auf Thermodynamik/Statistik (5. Semester) (unter Verwendung des Virialsatzes):

$$P\mathcal{V} = Nk_BT + \langle W \rangle$$

($\mathcal{V}$=Volumen) wobei das *Virial*

$$W = -\frac{1}{3}\sum_i \underline{r}_i \cdot \underline{\nabla}_i V$$

mit $\underline{f}_i := -\underline{\nabla}_i V$: Kraft auf Teilchen i.
Merke: Ohne Wecheslwirkung is $W = 0$, also $P\mathcal{V} = Nk_BT$ (ideales Gas).

Für Paarwechelwirkung, mit $f_{ij} = -f_{ji}$:

$$\begin{aligned}
W &= \frac{1}{3}\sum_i \underline{r}_i \cdot \underline{f}_i \\
&= \frac{1}{3}\sum_i \sum_{j\neq i} \underline{r}_i \cdot \underline{f}_{ij} \\
&= 0.5\left(\frac{1}{3}\sum_{i\neq j} \underline{r}_i \cdot \underline{f}_{ij} + \frac{1}{3}\sum_{i\neq j} \underline{r}_i \cdot \underline{f}_{ij}\right) \\
&= 0.5\left(\frac{1}{3}\sum_{i\neq j} \underline{r}_i \cdot \underline{f}_{ij} + \frac{1}{3}\sum_{i\neq j} \underline{r}_j \cdot \underline{f}_{ji}\right) \\
&= 0.5\frac{1}{3}\sum_{i\neq j} (\underline{r}_i - \underline{r}_j) \cdot \underline{f}_{ij} \\
&= \frac{1}{3}\sum_{i<j} (\underline{r}_i - \underline{r}_j) \cdot \underline{f}_{ij} \qquad (3)
\end{aligned}$$

Änderungen am Programm: Erweiterung von `glas_system_t` um

```
double          virial;       /* Virial = sum_(i>j) r_ij*f_ij/3 */
```

Erweiterung von `glas_energy_forces()` um Berechnung des Virials in der innersten Schleife gemäß (3)
Messung des Drucks als Funktion der Temperatur (Startkonfiguration dichteste Kugelpackung, 10000 Schritte, Druck = Mittelwert aus den letzten 5000 Schritte.):

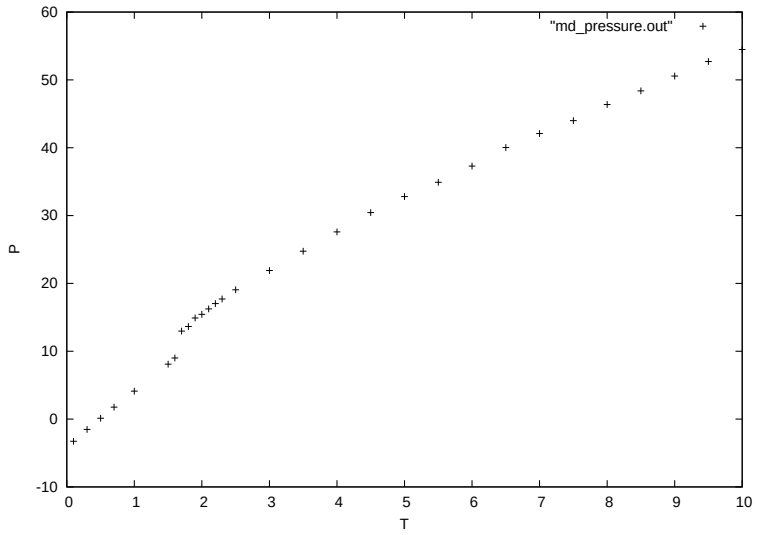


Abbildung 5: Druck als Funktion der Temperatur für LJ Gas.

Sprung zwischen $T = 1.6$ und $T = 1.7$: Phasenübergang kristalling → flüssig.

6.8 Abgeschnittene Potentiale

Berechnung der LJ-Wechselwirkung: Alle Paare von Atomen betrachtet → Rechenzeit $O(N^2)$, wird langsam für große Systeme.

Ausweg: Für große Abstände: Potential klein → "Abschneiden" für $r \geq r_{\text{cut}}$:

$$V_{LJ}^{\text{cut}}(r) = \begin{cases} V_{LJ}(r) & r < r_{\text{cut}} \\ 0 & r \geq r_{\text{cut}} \end{cases}$$

Typischerweise $r_{\text{cut}} = 2 - 3\sigma$.
Aber: Unstetigkeit bei $r = r_{\text{rc}}$ → Potential verschieben:

$$V_{LJ}^{\text{cut}}(r) = \begin{cases} V_{LJ}(r) - V_{LJ}(r_{\text{cut}}) & r < r_{\text{cut}} \\ 0 & r \geq r_{\text{cut}} \end{cases}$$

Allerdings: Kraft noch unstetig bei r_{cut}. Durch Interpolation mittels Splines auch Kraft stetig (wird aber meistens nicht gemacht).
Nun: Nur noch Atompaare mit $r < r_{\text{cut}}$ zu berücksichtigen.
Dazu: unterteilung des Systems in d-dim Boxen (*Verlet Boxen*), mit Kantenlängen $l_i \geq r_{\text{cut}}$.

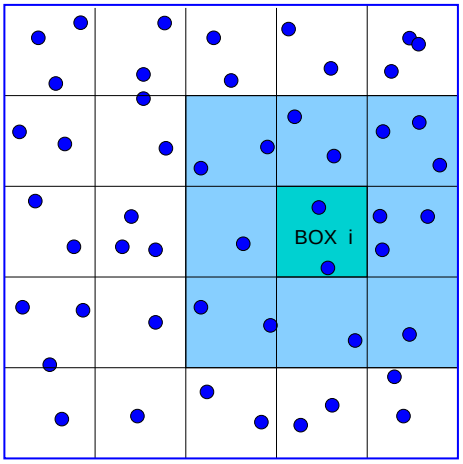


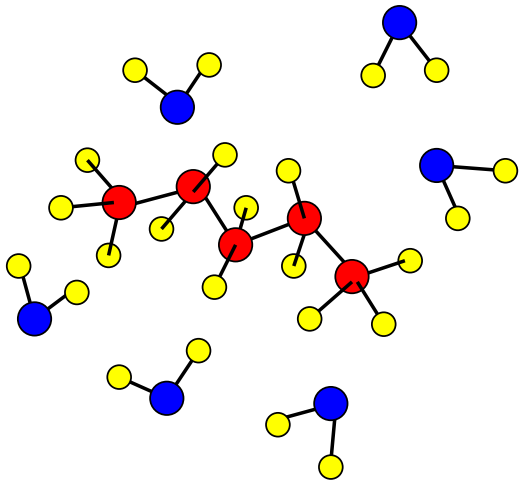
Abbildung 6: Bei kurzreichweitiger Wechselwirkung: Einteilung des Systems in Verlet Boxen → man braucht bei der Berechnung der Wechselwirkung nur über die eigene und die Nachbarboxen iterieren, statt über das ganze System.

Jede Box speichert die enthaltenen Atome. Wenn ein Atom die Box wechsel, müssen die Arrays geändert werden. Realisierung der Boxen als Arrays, sowie für jedes Atom die Boxnummer und seine Position in dem Box-Array speichern (in `gas_atom_t`). → Ein-/Ausfügen aus Box in Rechenzeit $O(1)$ erreichtbar.

Bei Berechnung der Kraft muß für jedes Atom nur über die eigene Box + alle Nachbarboxen iteriert werden → (bei konstanter Dichte) Gesamtrechenzeit $O(N)$!

6.9 Potentiale für Biomoleküle

Simulation von Biomolekülen = verschiedene geladene Polymeremoleküle = Ketten (mit Verzweigungen) von Atomen.



Entscheidend: Wahl des Potentials. Typisch: Gesamtpotential = gebundene Wechselwirkungen V_1 plus ungebundene (Fern-) Wechselwirkungen V_2:
Gebundene Wechselwirkungen: harmonische Bindungeslängen (Zwei-Teilchen WW) und Bindungswinkel (Drei-Teilchen WW), sowie Torsionswinkel (Vier-Teilchen WW)

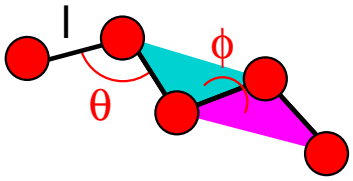


Abbildung 7: Gebundene Wechselwirkung: Bindungslänge l, Bindungswinkel θ und Torsionswinkel ϕ = Winkel zwischen denen durch je drei Atomen aufgespannten Ebenen.

$$V_1 = \sum_{\text{bonds}} k_l(l - l_0)^2 + \sum_{\text{angles}} k_\theta(\theta - \theta_0)^2 + \sum_{\text{torsions},n=1,2,3} A_n[1 + \cos(n\phi - \phi_0)]$$

Fernwechselwirkung: Lennard-Jones und Coulombwechselwirkung.

$$V_2 = V_{LJ}(r) + \sum_{i<j} \frac{q_i q_j}{4\pi\epsilon_r\epsilon_0 r_{ij}}$$

Coulomb-WW: langreichweitig → hoher Fehler beim Abschneiden: Besser: WW komplett mitnehmen (sog. *Ewald-Summen* bei periodischen Randbedingungen) oder hierarchische *Multipolentwicklung* (siehe Feldtheorie, 6. Semester).
Es gibt viele Teilchensorten → viele Parameter. Aus QM ab-initio Rechnungen / heuristischen Anpassungen bestimm.

Aufwändig: Biomomoleküle üblicherweise in Wasser gelöst → viele Wasserteilchen mitnehmen.