TERMODINÀMICA I MECÀNICA ESTADÍSTICA

Treball de simulació

Sofia Almirante, 1527718 Pau Reig, 1527816

Maig 2021

1 Gas d'esferes dures

Q1. Fixeu *N atoms* = 400 en el programa (i a partir d'ara treballeu sempre amb aquest valor). Feu algunes proves donant diferent valors al paràmetre *Ratom*. Discutiu quins problemes apareixen si un utilitza valors massa petits o massa grans d'aquest paràmetre, i en base a això escolliu quin valor de *Ratom* fareu servir a partir d'ara.

En primer lloc, donat que els objectes es troben confinats en un volum finit, és immediat trobar una cota superior per al valor del radi que poden tenir les partícules. Com que treballem amb un gas d'esferes dures, el volum total de les partícules, que imposem que ha de ser menor al volum del recipient que els conté, vindrà donat per:

$$V_N = NV_1 = N\frac{4}{3}\pi R_{atom}^3 < 1 \tag{1}$$

Aleshores, concloem que, com a màxim, el valor dels objectes ha de ser (per N=400):

$$R_{atom} < \sqrt[3]{\frac{3}{4\pi N}} = 0.084m \tag{2}$$

Per altra banda, hem de tenir en compte que estem treballant amb un gas d'esferes dures, on les partícules tenen un volum definit i no es poden solapar, doncs per resoldre un sistema on les partícules es comporten com un gas ideal no necessitem eines computacionals. Per aquest motiu no té sentit treballar amb un volum que sigui negligible respecte el volum del recipient. També cal afegir que les partícules difícilment col·lisionarien, cosa que comportaria que el sistema necessités evolucionar durant massa temps per assolir l'equilibri.

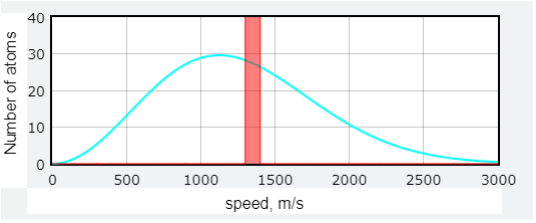

Figura 1: Distribució de velocitats del gas després de 10 minuts per un R_{atom}= 0.001 m

Per aquests motius, d'ara en endevant utilitzarem el valor $R_{atom} = 0.03m$, una mida prou gran com per a que es puguin produïr col·lisions entre les partícules i arribar a l'equilibri del sistema en un temps raonable, i també prou petita com per a que es puguin moure amb llibertat i compleixin la desigualtat (2).

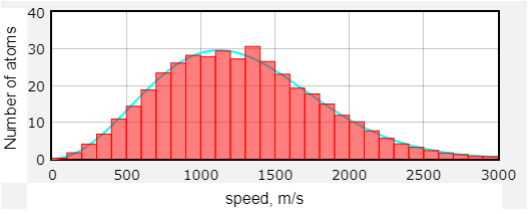

Figura 2: Distribució de velocitats del gas després de 10 minuts per un radi = 0.03

Q2. Modifiqueu el codi del programa per tal que la condició inicial correspongui a una situació d'equilibri tèrmic amb T = 300 K. Comproveu que el canvi funciona i adjunteu les línies de codi que heu fet servir per implementar-lo.

Quan un sistema termodinàmic es troba en equilibri, les components de les velocitats de les seves partícules satisfan la distribució de Maxwell-Boltzmann.

El mètode que emprarem per imposar que el sistema en inicialitzar-se es trobi en equilibri serà la transformada de Box-Muller.

Aquest mètode ens permet trobar un valor aleatori z_1 segons una distribució normal (gaussiana) amb $\sigma^2 = 1$ a partir de dos números aleatoris u_1 u_2 uniformement distribuits entre 0 i 1, mitjançant la següent expressió:

$$z_1 = \sqrt{-2\ln(u_1)}\cos(2\pi u_2) \tag{3}$$

Per últim, per tal que z_1 segueixi una distribució de Maxwell-Boltzmann caldrà multiplicar-lo per $\sigma = \sqrt{\frac{kT}{m}}$.

Amb tot això, les línies de codi on es definien els moments inicials son:

```
for i in range(Natoms):
```

```
        vx = sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)

        vy = sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)

        vz = sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)

        vmodul.append(sqrt(vx*vx+vy*vy+vz*vz))

        p.append(vector(vx*mass,vy*mass,vz*mass))

for i in range (nhisto): histo.append(0.0)
for i in range(Natoms):
    v = vmodul[i]
    histo[barx(v)] += 1
```

Q3. Modifiqueu el codi del programa de manera que us permeti calcular el coeficient de dilatació tèrmica del gas. Expliqueu com ho heu fet i mostreu els resultats obtinguts per a diferents valors de la pressió, i comproveu si s'obtenen els resultats teòrics que cal esperar per un gas ideal.

El coeficient de dilatació tèrmica del gas es defineix com:

$$\alpha = \frac{1}{V}\left(\frac{\partial V}{\partial T}\right)_P \tag{4}$$

Donat que la pressió és una magnitud que té molt soroll en el temps (ja que a cada instant pot prendre valors molt diferents), per treballar amb valors concrets de la pressió hem fet mitjanes temporals.

Per mantenir la pressió constant, la nostra estratègia ha estat la següent: En primer lloc, definim un rang de pressions $P \pm dP$ en el que volem que es mantingui el sistema.

Després anem augmentant la temperatura, deixant "reposar"el sistema $1000 \cdot dt$ *segons* abans de cada augment (per poder calcular la pressió fent la mitjana), i només augmentarà la temperatura si la pressió del sistema està en el nostre rang definit. Si $P \leq P - dT$ llavors disminuïm el volum $(V - dV)$ i si en canvi $P \geq P + dP$ augmentem el volum $(V + dV)$. Per tant, anotant cada cop que canviem la temperatura el volum al que estem, tenim la variació del volum en funció de la temperatura per un rang de pressions fixades. Si ho fem per dP prou petits, s'aproxima a la corba $V(T)$ a pressió constant.

Per a 3 pressions diferents s'han trobat les següents corbes $V(T)$:

Taula 1: Rectes V(T) a diferents pressions

P $(10^{-17}$**Pa)**	**V(T)**
1,25	0,00097T + 0,23900
1	0,00128T + 0,25773
0,75	0,00133T + 0,39487

Un cop conegudes les expressions d'aquestes rectes, aplicant l'expressió (4) podem trobar $\alpha(T)$ per cada pressió, per tal de poder comparar-ho amb el comportament ideal ($\alpha = \frac{1}{T}$). El comportament d'aquestes es veu representat a la Figura 3

Taula 2: $\alpha(T)$ Per diferents pressions

P $(10^{-17}$**Pa)**	$\alpha(T)$
1,25	$\frac{1}{T+247}$
1	$\frac{1}{T+202}$
0,75	$\frac{1}{T+296}$

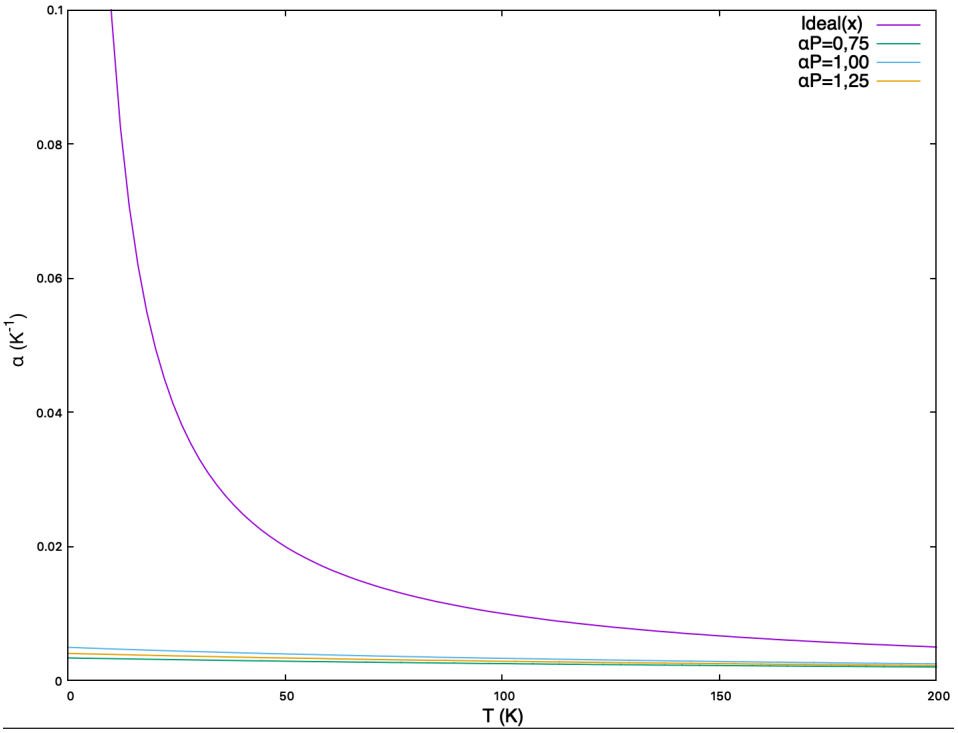

Figura 3: Diferents corbes del coeficient α per diferents valors de P i en lila el coeficient α del gas ideal

Podem veure que tot i allunyar-se del comportament ideal per temperatures baixes i apropar-se a mesura que augmenta la temperatura, les tres corbes d'α obtingudes per les 3 pressions diferents

són molt semblants. Això ens diu que, igual que passa per un gas ideal, en el sistema estudiat α no depèn en P.

També podem observar que té un valor aproximadament uniforme per totes les temperatures:

$$\alpha_{ct} \simeq 0,004 \ \ K^{-1} \tag{5}$$

Q4. Modifiqueu el comportament d'una de les parets de la caixa de tal manera que puguem reproduir en el gas una expansió lliure contra el buit. Expliqueu com ho heu fet i comproveu si la variació de la pressió amb el volum concorda amb el comportament teòric que cal esperar en un gas ideal.

Per fer l'expansió contra el buit augmentarem la longitud d'una de les parets de manera esglaonada deixant un interval de temps entre cada allargament per poder calcular la pressió corresponent fent la mitjana temporal.

Ho fem així perquè en una expansió contra el buit, mentre s'està produint, no es poden definir variables termodinàmiques (el sistema no està en equilibri). Per tant només podem obtenir informació del sistema abans i després de l'expansió, un cop el gas es troba en equilibri. Per aquest motiu fem vàries expansions contra el buit seguides (i petites, perquè arribi ràpid a l'equilibri) i així podem tenir una relació $P(V)$ i comparar-la amb un gas ideal.

A priori podem veure que la temperatura del sistema no variarà (igual que en un gas ideal) ja que la temperatura depèn de la mitjana de les velocitats i aquesta serà constant durant tot el procés degut a que tots els xocs entre partícules i parets són elàstics.

Seguint aquest procediment trobem la següent corba $P(V)$:

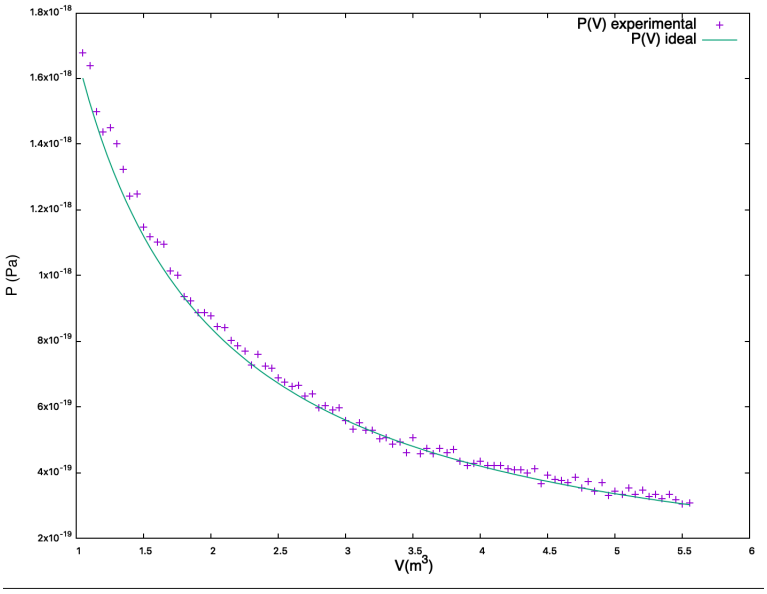

Figura 4: Comparació gas ideal i corba obtinguda a partir de l'expansió contra el buit del gas d'esferes dures

Ajustant els punts experimentals a una funció del tipus $f(x) = \frac{a}{x} + b$ ens queda:

$$P(V) = \frac{1,78071 \cdot 10^{-18}}{V} - 1,71934 \cdot 10^{-20} \tag{6}$$

Resultat molt semblant al del cas ideal (T=300 K, N = 400):

$$P(V)_{ideal} = \frac{NkT}{V} = \frac{1,68 \cdot 10^{-18}}{V} \tag{7}$$

2 Fluid de Lennard-Jones

Q5. Assegurant-vos que esteu a l'opció NVT, feu el següent: (i) Correu la simulació amb N = 500, = 0.8, T = 0.2 i observeu l'estat estacionari assolit. (ii) Correu una altra simulació durant uns pocs segons amb els valors N = 500, = 0.8 i T = 1.5, poseu la simulació en pausa, modifiqueu la temperatura a T = 0.2 i continueu la simulació fins a assolir un estat estacionari. Comenteu les diferències trobades en l'estat estacionari entre els casos (i) i (ii), tot discutint l'estabilitat en cada cas.

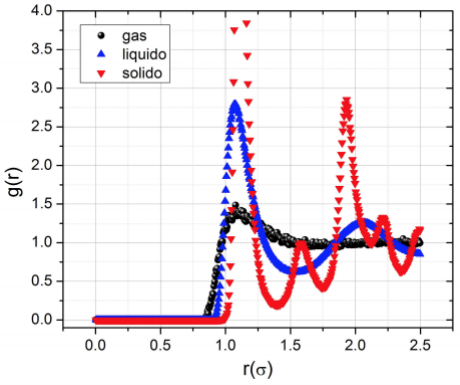

Figura 5: Funció de distribució radial en funció de la distància per un potencial de Lennard-Jones[2]

En inicialitzar el sistema amb T = 0.2, observem que l'estat estacionari que s'assoleix és el mostrat a la figura 6.

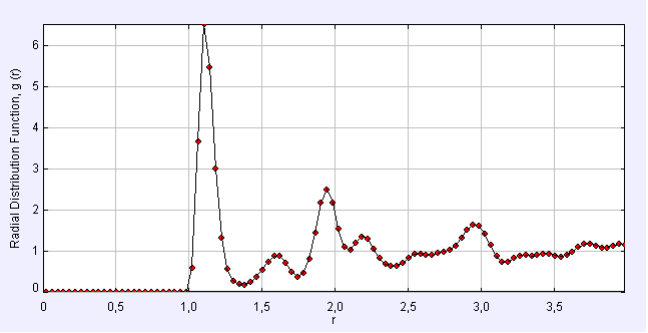

Figura 6: Distribució radial per a T = 0.2

Comparant la funció de distribució radial en funció de la distància amb el perfil teòric (Figura 5), podem concloure que es tracta d'un estat estable en fase sòlida. Donat que els pics es troben a valors enters de la distància, podem concloure que es tracta d'un sòlid cristal·lí.

En inicialitzar el sistema amb T = 1.5, en canvi, trobem el perfil que es veu a la figura 8. Podem concloure, en aquest cas, que es tracta d'un estat en fase líquida.

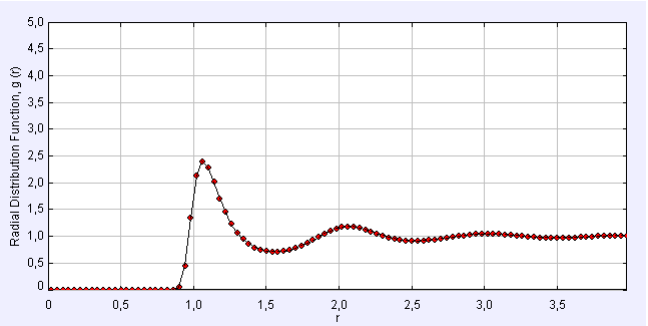

Figura 7: Distribució radial per a T = 1.5

En refredar ràpidament aquest sistema a T = 0.2, obtenim la distribució radial de la Figura 7, que podria concordar amb la d'un sòlid amorf. Aquest estat és metastable, doncs l'energia és superior a la que trobem en l'estat del cas (i), i en variar lleugerament la temperatura, el sistema decau al mateix.

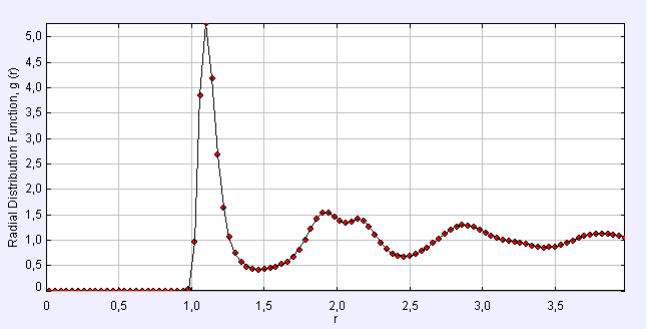

Figura 8: Distribució radial per a T = 0.2 després d'haver assolit l'equilibri per a T = 1.5

Q6. Utilitzeu el programa per a estudiar el comportament del coeficient de compressibilitat isoterma del gas en funció de la temperatura. Indiqueu en quina col·lectivitat treballeu, i compareu els resultats obtinguts amb els que s'obtindrien per (a) un gas ideal, i (b) un gas de Lennard-Jones (en aquest cas utilitzeu l'aproximació del 2n coeficient del virial B2 = 0.0445T + 7.274 14.34 sqrt T + 3.840/T 2.006/T2.

El coeficient de compressibilitat isoterma k_T és definit com:

$$k_T = -\frac{1}{V}\left(\frac{\partial V}{\partial p}\right)_T \tag{8}$$

Per aquest motiu, treballarem en la col·lectivitat NPT, que ens permet fixar la pressió i temperatura. D'aquesta manera, podrem obtenir de manera directa informació sobre el comportament del volum en funció de la pressió a una temperatura fixada. Les dades obtingudes es troben a la Taula 3.1.1 de l'Annex. En representar gràficament aquestes dades, observem que podem ajustar el comportament de V(p) a una funció del tipus $V(p) = \frac{a}{p} + b$, obtenint els valors següents:

Taula 3: Valors dels paràmetres a i b en ajustar els valors obtinguts a una funció del tipus $f(x) = \frac{a}{x} + b$

	T = 3	**T = 4**	**T = 5**	**T = 6**	**T = 7**	**T = 8**
a	1266 ±29	1821 ±17	2359 ±13	2904.0 ±7.7	3464.6 ±5.5	3985.6 ±5.6
b	454 ±32	478 ±13	489.1 ±7.1	493.4 ±4.2	487.0 ±5.3	488.8 ±5.6

Derivant les expressions V(p) obtingudes, obtenim el valor de k_T en funció de la pressió per a cada temperatura:

Les corbes obtingudes es troben representades a la Figura 9.

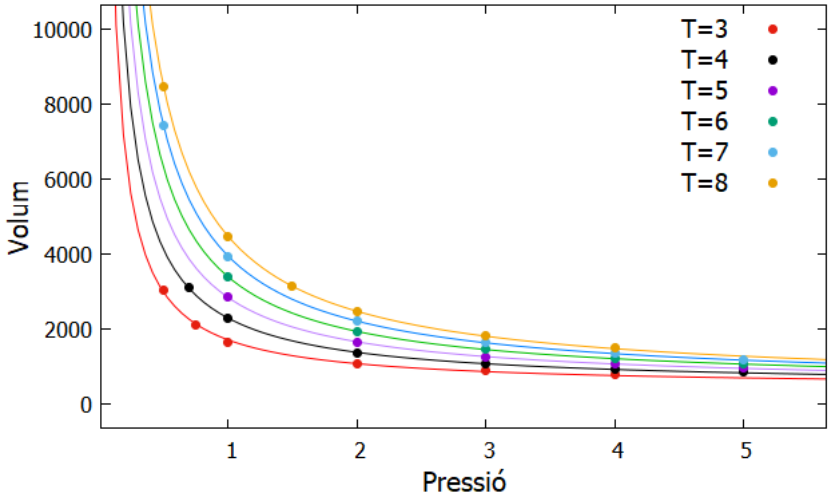

Figura 9: Corbes obtingudes del volum en funció de la pressió per a diferents temperatures.

$$k_T = \frac{-1}{V}\left(\frac{\partial\left(\frac{a}{V}+b\right)}{\partial p}\right)_T = \frac{1}{p + \frac{bp^2}{a}} < \frac{1}{p} \tag{9}$$

És immediat observar que aquest valor serà més petit que $k_T = \frac{1}{p}$ d'un gas ideal (donat que, per les temperatures triades, $a, b > 0$). D'altra banda, observem que b és aproximadament constant, mentre que a augmenta amb la temperatura. Així, al límit d'altes temperatures, observem que $k_{T_{sim}} \to k_{T_{id}}$. Aquest comportament es pot observar en les Figures 10 i 11, i pot ser degut a que el potencial de Lennard-Jones té més rellevància a baixes temperatures, on l'energia d'interacció entre partícules no és negligible respecte l'energia termodinàmica del sistema.

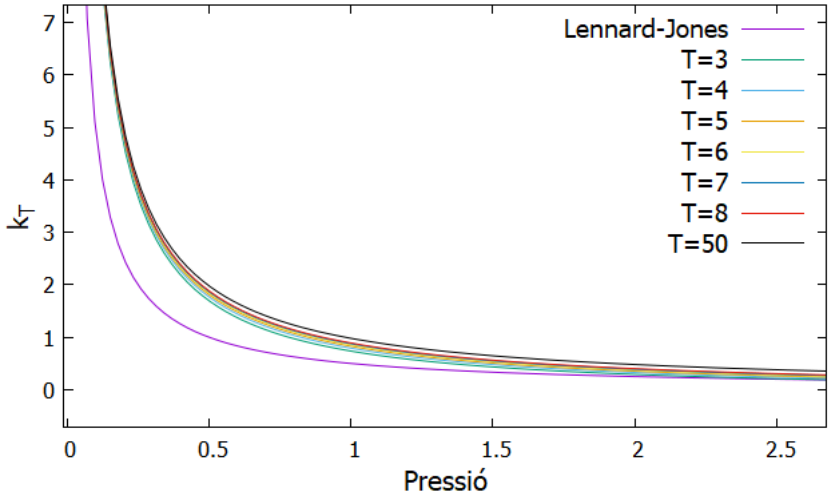

Figura 10: Coeficient k_T en funció de la pressió per a diverses temperatures en variables reduïdes, i comportament per un gas ideal.

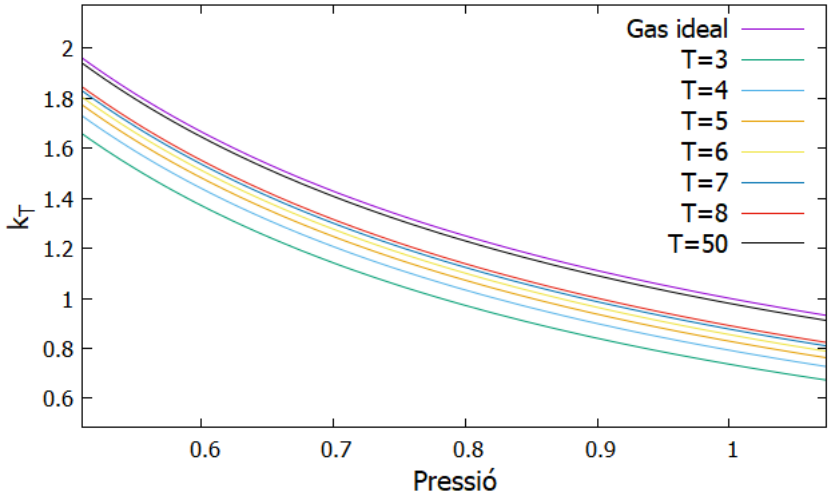

Figura 11: Ampliació de la Figura 10, on es mostra la tendència al gas ideal en augmentar la temperatura

Per altra banda, pel cas d'un fluid de Lennard-Jones, es compleix que:

$$F = -Nk_BT + \mu N + \frac{k_BTN^2B_2(T)}{V} \tag{10}$$

Sabent que $p = -\frac{\partial F}{\partial V}$, obtenim que:

$$p = \frac{k_BTN^2B_2(T)}{V^2} \tag{11}$$

Aleshores,

$$k_T = -\frac{1}{V}\left(\frac{\partial V}{\partial p}\right)_T = \frac{1}{2p} \tag{12}$$

A la Figura 12 podem observar com, a diferència del que passava amb el gas ideal, el gas del sistema estudiat tendeix a comportar-se com un fluid de Lennard-Jones per a baixes temperatures, així com per a altes pressions.

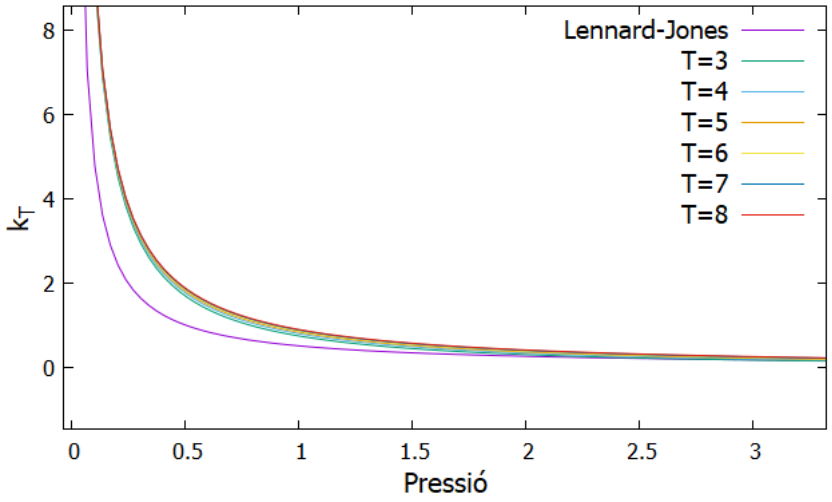

Figura 12: Coeficient k_T en funció de la pressió per a diverses temperatures en variables reduïdes, i comportament per un fluid de Lennard-Jones.

Hem realitzat aquest estudi independentment del valor del coeficient $B_2(T)$. Això és degut a que el $B_2(T)$ proporcionat per l'enunciat és sempre negatiu, per la qual cosa la pressió segons l'equació (10) resultaria negativa.

3 Transicions líquid-gas

Q7. Feu córrer la simulació amb els paràmetres per defecte excepte N1 = N2 = 256, i activant totes les opcions de output per poder visualitzar el màxim d'informació. Aneu modificant el valor de la temperatura i observant què passa, fins a construir la corba de coexistència en l'espai P-T, comparant-la amb la teòrica ln P = 3.063 6.756/T. Justifiqueu físicament què succeeix per T > 1.33.

Amb els valors obtinguts computacionalment que es troben a la Taula 5 de l'Annex, s'ha construït la corba de la Figura 13.

Ajustant els punts obtinguts a una funció del tipus $f(x) = \frac{a}{x} + b$, s'ha obtingut la corba $ln(p) = -(6.600 \pm 0.055)/T + (2.939 \pm 0.056)$, valors compatibles amb la corba teòrica proporcionada per l'enunciat.

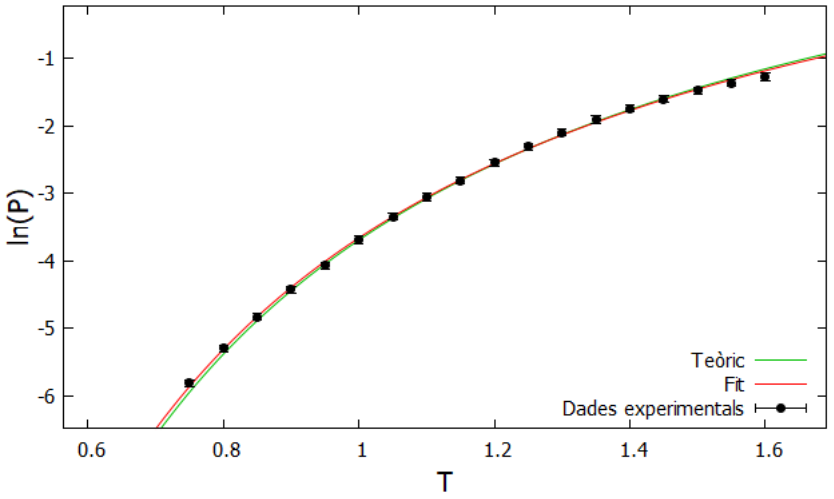

Figura 13: Construcció de la corba de coexistència ln(p)-T a partir de les dades obtingudes computacionalment i corba teòrica

Les dades s'han pres en intervals de temperatura de 0.5 un cop el sistema assoleix l'estat estacionari. S'ha observat, a mida que augmentava la temperatura, que tant la pressió com el potencial químic de les dues fases augmentava, sent igual per ambdues. Les diferència entre les densitats de les fases disminuïa , fent-se nul·la a partir de la temperatura crítica $T_c = 1.33$. S'ha observat, també, que a partir d'aquest punt les propietats termodinàmiques de les dues fases son idèntiques i, per tant, ambdues fases son indistingibles. Aquest fet també es pot apreciar visualment comparant les figures 14 i 15.

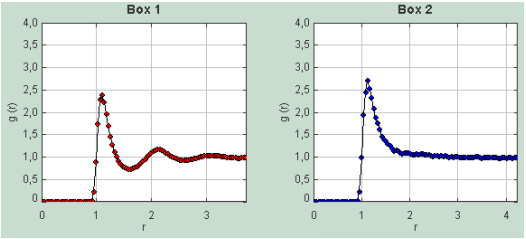

Figura 14: Funció de distribució radial de les dues fases, distingibles per a T=1

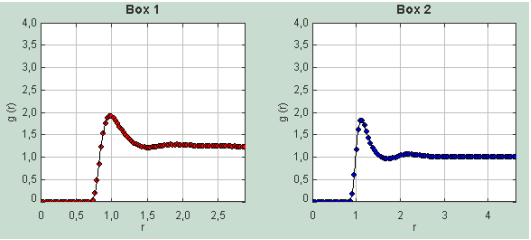

Figura 15: Funció de distribució radial de les dues fases, indistingibles per a T=1.4

Q8. Al llarg de la coexistència líquid-gas cal esperar un comportament del tipus $\rho_l - \rho_g = A(1 - T/T_c)^\beta$, on ρ_l, ρ_g són les densitats de cada una de les fases, T_c és la temperatura crítica, β és el paràmetre d'ordre, i A és una constant positiva. Utilitzeu les simulacions per tal de determinar els valors de β i A.

Amb els valors obtinguts computacionalment que es troben a la Taula 4 de l'Annex, s'ha construït la corba de la Figura 16.

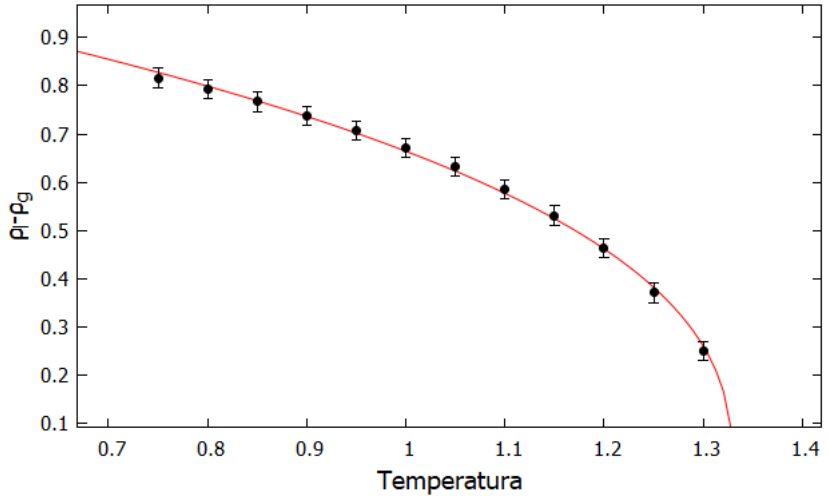

Figura 16: Diferència de densitats de les fases en funció de la temperatura

Ajustant aquests valors a una funció del tipus $f(x) = a(1 - \frac{x}{1.33})^b$, s'ha obtingut la funció $f(x) = (1.144 \pm 0.011)(1 - \frac{x}{1.33})^{(}0.3900 \pm 0.0068)$, d'on podem extreure el valor dels paràmetres A $= (1.144 \pm 0.011)$ i $\beta = (0.3900 \pm 0.0068)$.

Podem observar que el valor de β obtingut difereix lleugerament del valor estimat per altres autors a la literatura: 0.325 (Mick et al. (2013)[2], Okumura-Yonezawa(2000) [3])) o 0.36 ((Heyes (2015) [1]).

Annex

3.1 Valors obtinguts computacionalment

3.1.1 Fluid de Lennard-Jones

Taula 4: Valors del volum en variables reduïdes per a diferents valors de pressió i temperatura (també en variables reduïdes)

	T = 3	**T = 4**	**T = 5**	**T = 6**	**T = 7**	**T = 8**	**T = 50**
P = 0.5	3031.1	-	-	-	7422.1	8465.8	-
P = 0.75	2105.1	3096.8	-	-	-	-	-
P = 1	1667.1	2277.6	2852.7	3400.1	3940.45	4465.1	25558.2
P = 2	1078.9	1370.4	1656.8	1938.6	2211.9	3139.7	13046
P = 3	895.925	1084.1	1272.8	1460.57	1641.7	2477.9	8868.6
P = 4	801.98	941.5	1082.2	1219.9	1358.1	1822.1	6676.8
P = 5	-	854.95	967.3	1078.8	1187.38	1493.81	-

3.1.2 Transicions líquid-gas

Taula 5: Valors de la pressió i la densitat de les fases líquida i gas per a diferents temperatures (en variables reduïdes)

Temperatura	**Pressió gas**	**Pressió liquid**	ρ **gas**	ρ **liquid**
0.75	0.003	0.001	0.005	0.821
0.8	0.005	0.005	0.007	0.8
0.85	0.008	0.008	0.010	0.777
0.9	0.012	0.012	0.015	0.753
0.95	0.017	0.017	0.021	0.728
1	0.025	0.026	0.030	0.701
1.05	0.035	0.036	0.041	0.673
1.1	0.047	0.047	0.056	0.642
1.15	0.060	0.060	0.073	0.604
1.2	0.078	0.08	0.103	0.566
1.25	0.099	0.099	0.144	0.515
1.3	0.122	0.122	0.2	0.45
1.35	0.148	0.148	0.295	0.335
1.4	0.174	0.174	0.304	0.305
1.45	0.201	0.201	0.302	0.304
1.5	0.227	0.227	0.301	0.303
1.55	0.254	0.254	0.301	0.302
1.6	0.279	0.28	0.301	0.302

3.2 Codi

3.2.1 Q3

```
from vpython import *
import math
import numpy as np
#GlowScript 3.1 VPython

# Hard-sphere gas.

# Bruce Sherwood
f = open('P3_2(P=1.25).txt', 'w')

win = 500

Natoms = 400  # change this to have more or fewer atoms

Pressure = 1.25E-18
PressCount = 0
dV = 0.001
dP = 1E-20
volume = []
time = 0

# Typical values
L = 1 # container is a cube L on a side
gray = color.gray(0.7) # color of edges of container
mass = 4E-3/6E23 # helium mass
Ratom = 0.03 # wildly exaggerated size of helium atom
k = 1.4E-23 # Boltzmann constant
T = 150 # around room temperature
dt = 1E-5
dT = 5
SumPressure = 0


animation = canvas( width=win, height=win, align='left')
animation.range = L
animation.title = 'A "hard-sphere" gas'
s = """  Theoretical and averaged speed distributions (meters/sec).
  Initially all atoms have the same speed, but collisions
  change the speeds of the colliding atoms. One of the atoms is
  marked and leaves a trail so you can follow its path.

"""
```

Figura 17: línies 1-42

```
animation.caption = s
d = L/2+Ratom
r = 0.005
boxbottom = curve(color=gray, radius=r)
boxbottom.append([vector(-d,-d,-d), vector(-d,-d,d), vector(d,-d,d), vector(d,-d,-d), vector(-d,-d,-d)])
boxtop = curve(color=gray, radius=r)
boxtop.append([vector(-d,d,-d), vector(-d,d,d), vector(d,d,d), vector(d,d,-d), vector(-d,d,-d)])
vert1 = curve(color=gray, radius=r)
vert2 = curve(color=gray, radius=r)
vert3 = curve(color=gray, radius=r)
vert4 = curve(color=gray, radius=r)
vert1.append([vector(-d,-d,-d), vector(-d,d,-d)])
vert2.append([vector(-d,-d,d), vector(-d,d,d)])
vert3.append([vector(d,-d,d), vector(d,d,d)])
vert4.append([vector(d,-d,-d), vector(d,d,-d)])
Atoms = []
p = []
apos = []
vmodul = []
#pavg = sqrt(2*mass*1.5*k*T) # average kinetic energy p**2/(2mass) = (3/2)kT

for i in range(Natoms):
    x = L*random()-L/2
    y = L*random()-L/2
    z = L*random()-L/2
    if i == 0:
        Atoms.append(sphere(pos=vector(x,y,z), radius=Ratom, color=color.cyan, make_trail=True, retain=100,
    else: Atoms.append(sphere(pos=vector(x,y,z), radius=Ratom, color=gray))
    apos.append(vec(x,y,z))
    theta = pi*random()
    phi = 2*pi*random()
    #px = pavg*sin(theta)*cos(phi)
    #py = pavg*sin(theta)*sin(phi)
    #pz = pavg*cos(theta)
    #p.append(vector(px,py,pz))
for i in range(Natoms):

    vx = sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)

    vy = sqrt(-2*log(random()))*math.cos(2*pi*random())*ma
```

Figura 18: línies 43-82

```
84      vz = sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)
85
86      vmodul.append(sqrt(vx*vx+vy*vy+vz*vz))
87
88      p.append(vector(vx*mass,vy*mass,vz*mass))
89
90
91  deltav = 100 # binning for v histogram
92
93  def barx(v):
94      return int(v/deltav) # index into bars array
95
96  nhisto = int(10000/deltav)
97  histo = []
98  for i in range (nhisto): histo.append(0.0)
99  for i in range(Natoms):
00      v = vmodul[i]
01      histo[barx(v)] += 1
02
03
04  #histo[barx(pavg/mass)] = Natoms
05
06  gg = graph( width=win, height=0.4*win, xmax=3000, align='left',
07      xtitle='speed, m/s', ytitle='Number of atoms', ymax=Natoms*deltav/1000)
08
09  theory = gcurve( color=color.cyan )
10  dv = 10
11  for v in range(0,3001+dv,dv):  # theoretical prediction
12      theory.plot( v, (deltav/dv)*Natoms*4*pi*((mass/(2*pi*k*T))**1.5) *exp(-0.5*mass*(v**2)/(k*T))*(v**2)*dv
13
14  accum = []
15  for i in range(int(3000/deltav)): accum.append([deltav*(i+.5),0])
16  vdist = gvbars(color=color.red, delta=deltav )
17
18  def interchange(v1, v2):  # remove from v1 bar, add to v2 bar
19      barx1 = barx(v1)
20      barx2 = barx(v2)
21      if barx1 == barx2:  return
22      if barx1 >= len(histo) or barx2 >= len(histo): return
23      histo[barx1] -= 1
```

Figura 19: línies 84-123

```
124     histo[barx2] += 1
125
126 def checkCollisions():
127     hitlist = []
128     r2 = 2*Ratom
129     r2 *= r2
130     for i in range(Natoms):
131         ai = apos[i]
132         for j in range(i) :
133             aj = apos[j]
134             dr = ai - aj
135             if mag2(dr) < r2: hitlist.append([i,j])
136     return hitlist
137
138
139 nhisto = 0 # number of histogram snapshots to average
140 vol = 0
141 while T < 250:
142
143
144     T += dT
145     vol = 0
146     time = 0
147
148
149     while time <= 1100:
150
151
152         if SumPressure/100 <= Pressure - dP:
153             L -= dV
154         if SumPressure/100 >= Pressure + dP:
155             L += dV
156         if SumPressure/100 < Pressure + dP and SumPressure/100 > Pressure - dP:
157             volume.append(L*L*L)
158             vol +=1
159
160
161         if vol == 0 and time > 1000:
162             if SumPressure/100 < Pressure - dP:
163                 L -= 9*dV
```

Figura 20: línies 124-163

```
164             if SumPressure/100 > Pressure + dP:
165                 L += 9*dV
166 
167             time = 0
168 
169 
170 
171         SumPressure = 0
172 
173         for i in range(100):
174             rate(300)
175 
176             time += 1
177 
178             boxbottom.visible = False
179             boxtop.visible = False
180             vert1.visible = False
181             vert2.visible = False
182             vert3.visible = False
183             vert4.visible = False
184 
185             d = L/2+Ratom
186             r = 0.005
187             boxbottom = curve(color=gray, radius=r)
188             boxbottom.append([vector(-d,-d,-d), vector(-d,-d,d), vector(d,-d,d), vector(d,-d,-d), vector(-
189             boxtop = curve(color=gray, radius=r)
190             boxtop.append([vector(-d,d,-d), vector(-d,d,d), vector(d,d,d), vector(d,d,-d), vector(-d,d,-d)
191             vert1 = curve(color=gray, radius=r)
192             vert2 = curve(color=gray, radius=r)
193             vert3 = curve(color=gray, radius=r)
194             vert4 = curve(color=gray, radius=r)
195             vert1.append([vector(-d,-d,-d), vector(-d,d,-d)])
196             vert2.append([vector(-d,-d,d), vector(-d,d,d)])
197             vert3.append([vector(d,-d,d), vector(d,d,d)])
198             vert4.append([vector(d,-d,-d), vector(d,d,-d)])
199 
200             # Accumulate and average histogram snapshots
201             for i in range(len(accum)): accum[i][1] = (nhisto*accum[i][1] + histo[i])/(nhisto+1)
202             if nhisto % 10 == 0:
203                 vdist.data = accum
```

Figura 21: línies 164-203

```
205 
206             # Update all positions
207             for i in range(Natoms): Atoms[i].pos = apos[i] = apos[i] + (p[i]/mass)*dt
208 
209             # Check for collisions
210             hitlist = checkCollisions()
211 
212             # If any collisions took place, update momenta of the two atoms
213             for ij in hitlist:
214                 i = ij[0]
215                 j = ij[1]
216                 ptot = p[i]+p[j]
217                 posi = apos[i]
218                 posj = apos[j]
219                 vi = p[i]/mass
220                 vj = p[j]/mass
221                 vrel = vj-vi
222                 a = vrel.mag2
223                 if a == 0: continue;  # exactly same velocities
224                 rrel = posi-posj
225                 if rrel.mag > Ratom: continue # one atom went all the way through another
226 
227                 # theta is the angle between vrel and rrel:
228                 dx = dot(rrel, vrel.hat)       # rrel.mag*cos(theta)
229                 dy = cross(rrel, vrel.hat).mag # rrel.mag*sin(theta)
230                 # alpha is the angle of the triangle composed of rrel, path of atom j, and a line
231                 #   from the center of atom i to the center of atom j where atome j hits atom i:
232                 alpha = asin(dy/(2*Ratom))
233                 d = (2*Ratom)*cos(alpha)-dx # distance traveled into the atom from first contact
234                 deltat = d/vrel.mag         # time spent moving from first contact to position inside ato
235 
236                 posi = posi-vi*deltat # back up to contact configuration
237                 posj = posj-vj*deltat
238                 mtot = 2*mass
239                 pcmi = p[i]-ptot*mass/mtot # transform momenta to cm frame
240                 pcmj = p[j]-ptot*mass/mtot
241                 rrel = norm(rrel)
```

Figura 22: línies 205-241

```
250
251             for i in range(Natoms):
252                 loc = apos[i]
253                 if abs(loc.x) > L/2:
254                     PressCount += 2*abs(p[i].x)/(dt*L*L*6)
255                     if loc.x < 0:
256                         loc.x = -L/2
257                         p[i].x =  abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*ma
258                     else:
259                         loc.x = L/2
260                         p[i].x =  -abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*m
261
262                 if abs(loc.y) > L/2:
263                     PressCount += 2*abs(p[i].y)/(dt*L*L*6)
264                     if loc.y < 0:
265                         loc.y = -L/2
266                         p[i].y = abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*mas
267                     else:
268                         loc.y = L/2
269                         p[i].y =  -abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*m
270
271                 if abs(loc.z) > L/2:
272                     PressCount += 2*abs(p[i].z)/(dt*L*L*6)
273                     if loc.z < 0:
274                         loc.z = -L/2
275                         p[i].z =  abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*ma
276                     else:
277                         loc.z = L/2
278                         p[i].z =  -abs(sqrt(-2*log(random()))*math.cos(2*pi*random())*math.sqrt(k*T/mass)*m
279
280             SumPressure += PressCount
281
282             PressCount = 0
283
284     f.write('%lf   %lf   \n' %(T,sum(volume)/len(volume)))
285 f.close()
```

Figura 23: línies 250-285

3.2.2 Q4

```
148 while Ly < 10:
149     Ly += dV
150     Temps += 1
151     SumPressure = 0
152     for q in range(ENA):
153
154
155         rate(300)
156
157         boxbottom.visible = False
158         boxtop.visible = False
159         vert1.visible = False
160         vert2.visible = False
161         vert3.visible = False
162         vert4.visible = False
163
164         y = []
165
166         #for i in range(Natoms):
167             #y.append(apos[i].y)
168
169         #Ly = abs(min(y))
170
171
172
173         d = L/2+Ratom
174         d2 = Ly/2+Ratom
175         r = 0.005
176         boxbottom = curve(color=gray, radius=r)
177         boxbottom.append([vector(-d,-d2,-d), vector(-d,-d2,d), vector(d,-d2,d), vector(d,-d2,-d), vector(-d
178         boxtop = curve(color=gray, radius=r)
179         boxtop.append([vector(-d,d,-d), vector(-d,d,d), vector(d,d,d), vector(d,d,-d), vector(-d,d,-d)])
180         vert1 = curve(color=gray, radius=r)
181         vert2 = curve(color=gray, radius=r)
182         vert3 = curve(color=gray, radius=r)
183         vert4 = curve(color=gray, radius=r)
184         vert1.append([vector(-d,-d2,-d), vector(-d,d,-d)])
```

Figura 24: línies 148-184

```
240         for i in range(Natoms):
241             loc = apos[i]
242             if abs(loc.x) > L/2:
243                 Pressure += 2*abs(p[i].x)/(dt*(4*L*(Ly+L)*L/2 + 2*L*L))
244                 if loc.x < 0:
245                     p[i].x =  abs(p[i].x)
246                 else:
247                     p[i].x =  -abs(p[i].x)
248 
249             if loc.y > L/2:
250                 Pressure += 2*abs(p[i].y)/(dt*(4*L*(Ly+L)*L/2 + 2*L*L))
251                 p[i].y = -abs(p[i].y)
252 
253             if loc.y < -Ly/2:
254                 Pressure += 2*abs(p[i].y)/(dt*(4*L*(Ly+L)*L/2 + 2*L*L))
255                 p[i].y = abs(p[i].y)
256 
257             if abs(loc.z) > L/2:
258                 Pressure += 2*abs(p[i].z)/(dt*(4*L*(Ly+L)*L/2 + 2*L*L))
259                 if loc.z < 0:
260                     p[i].z =  abs(p[i].z)
261                 else:
262                     p[i].z =  -abs(p[i].z)
263 
264         SumPressure += Pressure
265         Pressure = 0
266         #Volumes.append(L*L*Ly)
267 
268 
269 
270     f.write('%lf  %.30lf  \n' %(L*L*(Ly+L)/2,SumPressure/ENA))
271 
272 f.close()
273 
274 
275 
276 
```

Figura 25: línies 240-276

Referències

[1] D. M. Heyes. *The lennard-jones fluid in the liquid-vapour critical region.* CMST,21(4):169–179, 2015.

[2] J. Mick, E. Hailat, V. Russo, K. Rushaidat, L. Schwiebert, and J. Potoff. *Gpu-accelerated gibbs ensemble monte carlo simulations of lennard-jonesium.*, Comp.Phys. Comm., 184(12):2662–2669, 2013.

[3] H. Okumura and F. Yonezawa. *Liquid–vapor coexistence curves of several inter-atomic model potentials.*J. Chem. Phys., 113(20):9162–9168, 2000.