Obligatorio 1:
Creación de la Red Inicial

González Rusell, Pablo

31 de octubre de 2022

Objetivos del programa:

1. Colocación de $N = 500$ partículas en una red fcc que ocupe completamente una caja cúbica de lado $L = 10$.
2. Desordenación (en cierto grado) de las partículas de su configuración anterior.
3. Cálculo de la energía potencial del sistema y las aceleraciones de las partículas a partir de la configuración anterior, supuesto un potencial *Lennar-Jonnes* $12 - 6$ truncado y aplicando condiciones de contorno periódicas.
4. Cálculo de las velocidades de las partículas y la energía cinética necesaria para que la energía total del sistema sea una dada, $E_{\text{tot}} = -690$.
5. A partir de la configuración inicial anteriormente creada ($\mathbf{r}$, $\mathbf{v}$, $\mathbf{a}$, E_{tot}, E_{cin}, E_{pot}), dejar que la configuración evolucione un tiempo $t = \Delta t \cdot N_{\text{pasos}}$, con $\Delta t = 10^{-4}$ y $N_{\text{pasos}} = 5000$, en nuestro caso.
6. Guardado en un archivo de las energías y el tiempo en cada uno de los pasos para su posterior tratamiento.

ATENCIÓN:
El presente documento trata de aclarar el funcionamiento, en rasgos generales, del programa, las aclaraciones concretas acerca de qué significa cada variable y cómo está definida se hacen en el propio programa, a modo de código comentado.

Índice

1. Creación de la red fcc

A la hora de construir nuestra configuración inicial debemos colocar las N partículas en algún lugar de la caja de volumen $V = \ell^3$. La opción por la que optamos aquí es por dividir dicha caja en celdas unitarias como la que muestra en la figura 1.

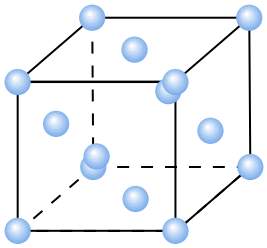

Figura 1: Celda unitaria convencional de la red fcc, con lado a. En ella las partículas ocupan los vértices y los centros de las caras.

Para que esta división pueda efectuarse, debe guardarse una relación entre ℓ y a, que básicamente es:

$$\ell = ka \Longrightarrow a = \frac{\ell}{k} \tag{1}$$

donde k tiene que ser un número entero. Evidentemente fijar k fija el número de partículas totales de la caja, en una celda fcc hay cuatro partículas netas, por tanto,

$$N = 4k^3 \Longrightarrow k = \sqrt[3]{\frac{N}{4}} \tag{2}$$

En nuetro caso, fijaremos $\ell = 10$, $N = 500$ y obtendremos a y k a partir de las ecuaciones 1 y 2, que programaremos del siguiente modo:

```
1 k=ceiling((N/4.d00)**(1.d00/3.d00),entero)
2 a=L/k
```

Notemos que, k tiene que ser un entero, en nuestro caso el resultado de la operación de la línea 1, (sin usar el ceil) era $4{,}9999999999999991$, por ello usamos la función ceil que convertía un real en el entero superior, en nuestro caso, 5.

La forma más sencilla de crear la red es construir una celda patrón e irla repitiendo iterativamente hasta cubrir toda la caja. Como celda patrón no nos sirve la de la figura 1, pues se producirían solapamientos, en su lugar vamos a usar la que se muestra en la figura 2

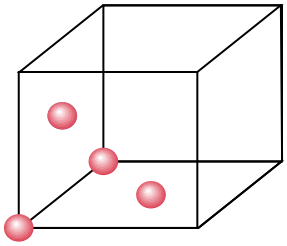

Figura 2: Celda patrón

Nótese que es totalmente equivalente a la celda de la figura 1 ya que contiene el mismo número de átomos por celda, pero en este caso pueden yuxtaponerse sin solaparse. A continuación asignamos las posiciones de las 4 primeras partículas a las correspondientes a la celda patrón:

```
rx(1)=0
rx(2)=a/2
rx(3)=a/2
rx(4)=0

ry(1)=0
ry(2)=a/2
ry(3)=0
ry(4)=a/2

rz(1)=0
rz(2)=0
rz(3)=a/2
rz(4)=a/2
```

Ahora debemos repetir este esquema $k-1$ veces en la dirección x.[1]:

```
do i=1,(k-1)
    do j=1,4
        m=4*i+j
        rx(m)=rx(j)+a*i
        rz(m)=rz(j)
        ry(m)=ry(j)
    end do
end do
```

Para colocar las partículas empleamos un bucle con dos contadores, j va a ir desde 1 hasta 4, ya que es el que reproduce el patrón inicial de 4 átomos. Mientras, el índice i es el que lleva cuenta de en que repetición estamos, yendo desde 1 hasta 4. Ahora para poder asignar las posiciones necesitaríamos un tercer contador que vaya desde la partícula 5 hasta la 20 (que son las que vamos a asignar en este primer paso), para ello usamos m que, por construcción va desde $4\times 1+1=5$ hasta $4\times 4+4=20$. En cuanto a la asignación, las posiciones x, y, z de las partículas son las mismas que las del patrón, salvo que x aumenta en una unidad de a por cada repetición.

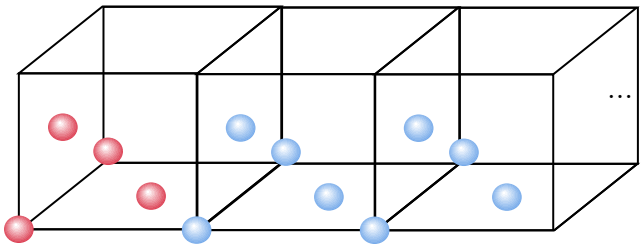

Figura 3: Repetición en x

El siguiente paso es repetir este proceso exactamente en el eje y pero cambiando el patrón, en lugar de ser las cuatro partículas de la primera celda, son las 20 partículas de la línea de cajas. El programa

[1] Debe ser $k-1$ porque si queremos llenar dos celdas en total debemos repetir 2-1 veces la primera.

es absolutamente análogo salvo en que j en vez de ir hasta 4, tiene que ir hasta 20 (para generalizar el programa para $k \neq 5$ guardamos el número de la última partícula del proceso anterior para que j corra hasta él) y ahora la posición que vamos aumentando es la y. Solo falta modificar m para que corra desde, 21 hasta 100:

```
p1=m
do i=1,(k-1)
     do j=1,p1
          m=p1*i+j
          rx(m)=rx(j)
          ry(m)=ry(j)+a*i
          rz(m)=rz(j)
     end do
end do
```

Y ahora tenemos una configuración siguiente:

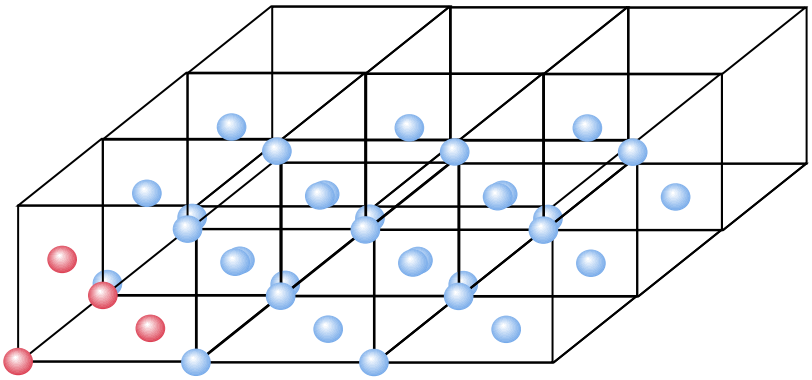

Figura 4: Repetición en y

Y ahora volvemos a hacer lo mismo que en el paso anterior, salvo que ahora el patrón a repetir son las 100 partículas, de ahí que $j = 1, ..., 100$ (donde 100 es la posición de la última partícula asignada en el paso anterior) pero aumentando z y m toma valores ahora entre 101 y 500.

```
p2=m
do i=1,(k-1)
     do j=1,p2
          m=p2*i+j
          rx(m)=rx(j)
          ry(m)=ry(j)
          rz(m)=rz(j)+a*i
     end do
end do
```

Y con este programa ya tendríamos la caja con las 500 partículas asignadas formando una red fcc. Para ver que, efectivamente lo hace, podemos grabar las posiciones de las partículas en un archivo y representarlas gráficamente. Como representar al representar 500 partículas no vemos nada vamos a hacer *zoom* y quedarnos solo con 32 de ellas:

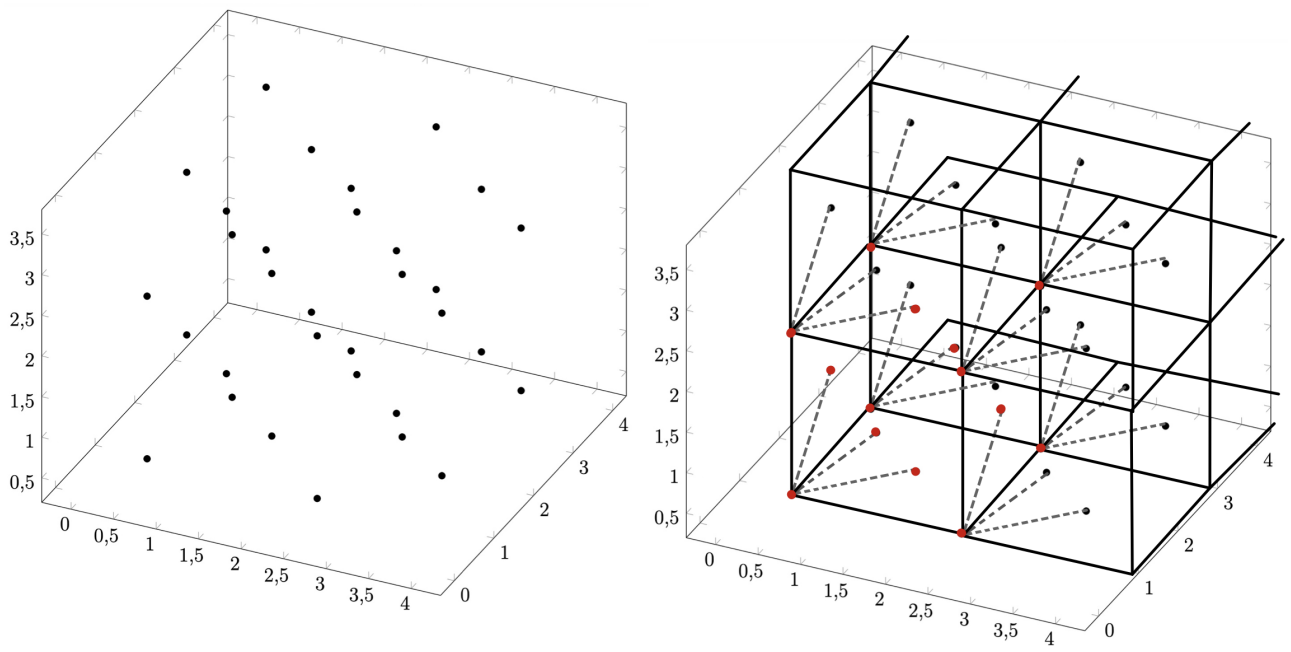

Figura 5: Representación gráfica de las posiciones de $N = 32$ partículas (izquierda) junto con el esqueleto de la red fcc (derecha)

Vemos, entonces, que nuestro programa coloca las partículas del modo deseado.

2. Desordenación aleatoria

Antes de comenzar a operar con el sistema, debemos ser conscientes de que, una configuración de tal alta simetría puede traer asociados problemas con la energía potencial que, puede provocar un pequeño salto en la energía total a la hora de hacer evolucionar temporalmente a nuestro sistema. Para evitarlo, podemos desplazar una cierta distancia aleatoria cada una de las partículas. Esa cierta distancia debe estar acotada ya que, en caso contrario podrían estar unas demasiado cerca de otras produciendo una divergencia de la energía. Evidentemente, como máximo debería ser un desplazamiento menos $a\sqrt{2}/4 \simeq 0,35a$, pues el vecino más próximo está a $a\sqrt{2}/2$. Sin embargo, en algunos casos las partículas podrían estar demasiado próximas las unas de las otras y disparar la energía también, de modo que, por ensayo y error, hemos visto que el mejor escenario es desplazar $(-a/8, a/8)$.

Para poder programarlo hacemos uso de la función mi_random() racilitada por el profesor, que toma un número entero que utilizará como *seed* y devolverá un número en el intervalo $[0, 1)$ cada vez que sea llamada. El programa es el siguiente:

```
do i=1,N

    rx(i)=rx(i)+b*(0.5-mi_random(iduma))+a/4
    ry(i)=ry(i)+b*(0.5-mi_random(iduma))+a/4
    rz(i)=rz(i)+b*(0.5-mi_random(iduma))+a/4

end do
```

Donde $b = a/4$, iduma un número entero inicializado al comienzo del programa. El término del

paréntiesis va desde $-0,5$ hasta $0,5$, luego estamos desplanzado en el intervalo $[-a/8, a/8)$, tal y como deseábamos. Además, el sistema inicialmente no tiene su centro de masas en el centro del cubo, ya que hay partículas en las caras que confluyen en el origen de coordenadas pero no en el vértice opuesto, para ello sumamos $a/4$ en cada dirección y así el centro de masas es el deseado.

A continuación representamos los puntos sin desplazar junto con los puntos de la configuración actual para ver el efecto:

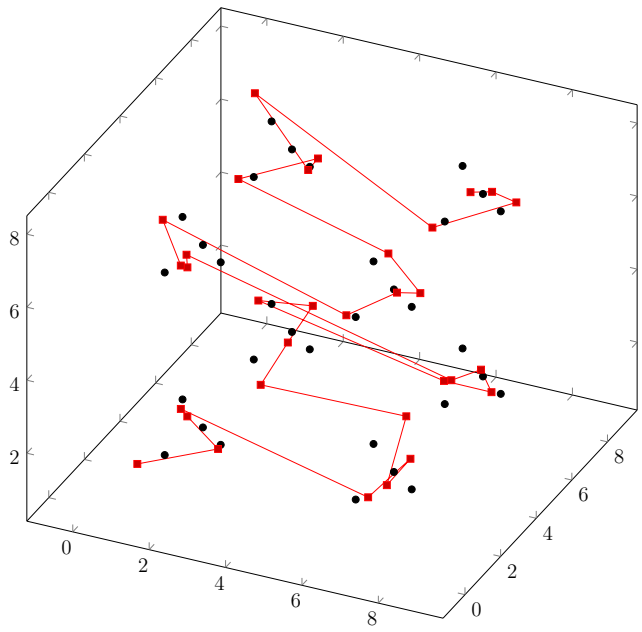

Figura 6: Representación de la red inicial y la red descolocada. Por comodidad a la hora de ver el resultado, hemos colocado $N = 32$ partículas en la caja completa (no es un *zoom*).

Efectivamente, observamos que, si bien la red ya no se parece en nada a un fcc, sí que la distancia entre los puntos nuevos y los originales parece ser la correcta, por lo que damos por bueno el programa.

3. Cálculo de la energía potencial

En esta simulación vamos a trabajar con un potencial par entre partículas correspondiente a un *Lennar-Johnes* $12-6$ truncado a un radio de corte (r_c), i.e,

$$v_{ij}(r_{ij}) = \begin{cases} 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] & \text{si} \quad r_{ij} \leq r_c \\ 0 & \text{si} \quad r_{ij} > r_c \end{cases} \tag{3}$$

con el que trabajaremos en unidades reducidas,

$$v_{ij}(r_{ij}) = \begin{cases} 4\left[\left(\frac{1}{r_{ij}}\right)^{12} - \left(\frac{1}{r_{ij}}\right)^{6}\right] & \text{si} \quad r_{ij} \leq r_c \\ 0 & \text{si} \quad r_{ij} > r_c \end{cases} \tag{4}$$

Y, por tanto, la energía potencial será,

$$E_p = \sum_{i=1}^{N-1} \sum_{i=j+1}^{N} v_{ij}(r_{ij}) \tag{5}$$

A partir del potencial, obtener la fuerza es relativamente sencillo siguiendo los pasos de los apuntes de la asignatura y llegando a que,

$$\mathbf{F}_k = \sum_{\substack{j=1 \\ j\neq k}}^{N} F_{kj}^{\text{mod}}\ \mathbf{r}_{kj} \tag{6}$$

donde

$$F_{kj}^{\text{mod}} = 24\left[2\left(\frac{1}{r_{ij}}\right)^{12} - \left(\frac{1}{r_{ij}}\right)^{6}\right]\frac{1}{r_{jk}^2} \tag{7}$$

Dada la simetría de F_{kj}^{mod} y la antisimetría $\mathbf{r}_{kj}$,

$$\sum_{k=1}^{N}\mathbf{F}_k = \sum_{k=1}^{N}\sum_{\substack{j=1 \\ j\neq k}}^{N} F_{kj}^{\text{mod}}\ \mathbf{r}_{kj} = 0 \tag{8}$$

Ya que tenemos contraídos índices simétricos con antisimétricos y, por tanto, la suma de las fuerzas es nula.

Para la obtención de otras magnitudes del sistema (en ejercicios posteriores) vamos a necesitar también las cantidades:

$$\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} r_{ij}v'_{ij}(r_{ij}) = 24\left[-2\left(\frac{1}{r_{ij}}\right)^{12} + \left(\frac{1}{r_{ij}}\right)^{6}\right] \tag{9}$$

$$\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} r_{ij}^2 v''_{ij}(r_{ij}) = 24\left[26\left(\frac{1}{r_{ij}}\right)^{12} - 7\left(\frac{1}{r_{ij}}\right)^{6}\right] \tag{10}$$

Sin embargo, es claro que el sistema físico real que nosotros queremos simular no va a seguir el potencial L-J truncado, en todo caso seguirá el real. Esto nos obliga a tener que corregir que alguna manera los valores de las energías y sus derivadas. Para ello vamos a emplear el formalismo de la función de distribución $g(r)$ que nos asegura que:

$$E_p = \frac{2\pi N^2}{V}\int_0^{\infty} v(r)g(r)r^2\,\mathrm{d}r \tag{11}$$

Y, por tanto, integrando,

$$E_p = \langle E_p\rangle + N u_{\text{LR}} \tag{12}$$

donde $\langle E_p\rangle$ es la energía potencial calculada como en 4 y donde,

$$u_{\text{LR}} = \frac{8\pi N}{3Vr_c^3}\left(\frac{1}{3r_c^6} - 1\right) \tag{13}$$

donde se ha usado $g(r) \simeq 0$. Por otro lado,

$$\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} r_{ij}v'_{ij}(r_{ij}) = \frac{2\pi N^2}{V}\int_0^{\infty} v'(r)g(r)r^3\mathrm{d}r \tag{14}$$

Calculemos ahora la integral,

$$\frac{2\pi N^2}{V}\int_0^{\infty} v'(r)g(r)r^3\mathrm{d}r = \frac{2\pi N^2}{V}\int_0^{r_c} v'(r)g(r)r^3\mathrm{d}r + \frac{2\pi N^2}{V}\int_{r_c}^{\infty} v'(r)g(r)r^3\mathrm{d}r =$$

$$= \left\langle \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} r_{ij} v'_{ij}(r_{ij}) \right\rangle + N\alpha$$

Donde α será la corrección por partícula al término de la derivada del potencial. Calculémosla,

$$\alpha = 2\pi\rho \int_{r_c}^{\infty} \left\{ 24 \left[-2 \left(\frac{1}{r}\right)^{10} + \left(\frac{1}{r}\right)^{4} \right] \right\} \mathrm{d}r = 48\pi\rho \left[\frac{2}{9} \left(\frac{1}{r}\right)^{9} - \frac{1}{3} \left(\frac{1}{r}\right)^{3} \right]_{r_c}^{\infty} =$$

$$\alpha = \frac{16\pi\rho}{r_c^3} \left(1 - \frac{2}{3r_c^6} \right) \tag{15}$$

Por otro lado,

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} r_{ij}^2 v''_{ij}(r_{ij}) = \frac{2\pi N^2}{V} \int_0^{\infty} v'' g(r) r^4 \, \mathrm{d}r \tag{16}$$

Que, siguiendo un razonamiento análogo al anterior conduce a una corrección por partícula,

$$\beta = 2\pi\rho \int_{r_c}^{\infty} 24 \left[26 \left(\frac{1}{r}\right)^{10} - 7 \left(\frac{1}{r}\right)^{4} \right] \mathrm{d}r = 48\pi\rho \left[-\frac{26}{9} \left(\frac{1}{r}\right)^{9} + \frac{7}{3} \left(\frac{1}{r}\right)^{3} \right]_{r_c}^{\infty}$$

$$\beta = \frac{16\pi\rho}{r_c^3} \left(\frac{26}{3r_c^6} - 7 \right) \tag{17}$$

Un último detalle es que, dado que etamos empleando un número de partículas relativamente bajo, no nos sirve con simular 500 partículas encerradas en una caja para obtener la termodinámica del sistema, lo ideal sería emplear condiciones periódicas de contorno, es decir, “rodear” nuestra caja por infinitas cajas idénticas, de modo que si una partícula sale por un lado, su imagen en la caja vecina entra por el otro. Véase figura 7

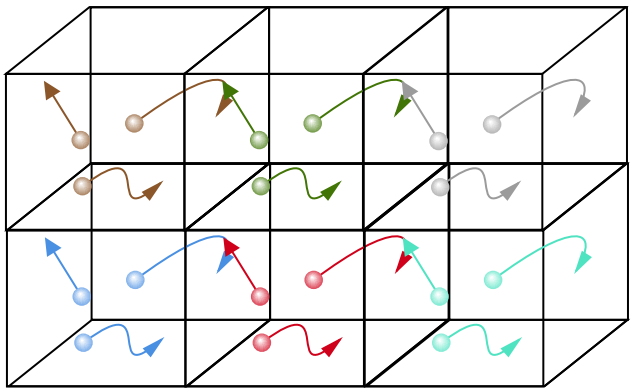

Figura 7: Representación esquemática de las condiciones periódicas de contorno

Sin embargo, no es técnicamente posible programar infinitas partículas alrededor de la nuestra, lo que vamos a hacer, simplemente, es calcular la imagen de la partícula más próxima a nuestra caja inicial. Es decir, lo que se muestra a continuación.

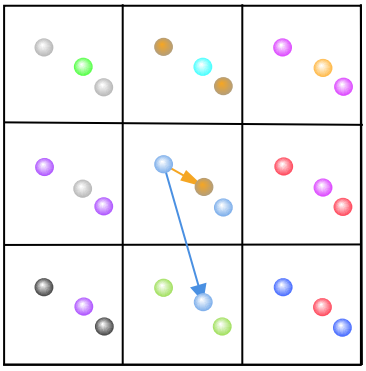

Figura 8: Configuración de una parte del sistema pasado un cierto tiempo. La celda central es nuestra celda de trabajo y el resto son las "copias". Las partículas de un mismo color se encontraban inicialmente en una misma copia.

En este ejemplo mostrado en la figura 8, a la hora de calcula el potencial entre la partícula azul y la partícula azul señalada con la flecha azul, lo haríamo tomando en cuenta la distancia con su partícula imagen más cercana, i.e, la señalada con la flecha naranja. Para calcular la posición exacta de la partícula naranja, primero dividimos la de la azul entre el lado de la caja. En este caso obtendríamos, por ejemplo, $1,32$, entonces tendremos que trasladar la partícula azul una unidad del lado de la caja.

A continuación se muestra el programa empleado:

```
do i=1,N-1

        rxi=rx(i)
        ryi=ry(i)
        rzi=rz(i)

        do j=i+1,N

            rxij=rxi-rx(j)
            ryij=ryi-ry(j)
            rzij=rzi-rz(j)

! Calculamos rij como la distancia entre la part cula i y la imagen m s
    pr xima de la part cula j

            rxij=rxij-L*dnint(rxij*L_inv)
            ryij=ryij-L*dnint(ryij*L_inv)
            rzij=rzij-L*dnint(rzij*L_inv)

            d2=rxij*rxij+ryij*ryij+rzij*rzij

             d2_inv=1/d2
             d6_inv=d2_inv*d2_inv*d2_inv
             d12_inv=d6_inv*d6_inv

! Si la imagen de la part cula j m s pr xima est  dentro del radio de
    corte estas interact an y calculamos
! la energ a potencial y la fuerza (aceleraci n, m=1) de la part cula
   i.

```

```
            if (d2<=rc2) then

! Sumamos a la energ a potencial, la de la part cula

                factor=-2*d12_inv+d6_inv
                epot=epot+d12_inv-d6_inv
                depotr=depotr+factor
                ddepotrr=26*d12_inv-7*d6_inv
                fmod=(-factor)*d2_inv

! Calculamos la fuerza

                ax(i)=ax(i)+fmod*rxij
                ax(j)=ax(j)-fmod*rxij
!
                ay(i)=ay(i)+fmod*ryij
                ay(j)=ay(j)-fmod*ryij
!
                az(i)=az(i)+fmod*rzij
                az(j)=az(j)-fmod*rzij
!
            end if
        end do
    end do

        epot=4*epot
        depotr=24*depotr
        ddepotrr=24*ddepotrr
        ax=24*ax
        ay=24*ay
        az=24*az
```

Y finalmente, añadimos las correcciones a la energía.

```
epot=epot+ecorr
depotr=depotr+depotr_corr
ddepotrr=ddepotrr+ddepotrr_corr
```

4. Asignación de velocidades y energía cinética

En esta simulación trabajamos en el conjunto NVE, de modo que la energía total es una constante del movimiento, en nuestro caso concreto vale, $E_{\text{tot}} = -690$. Una vez hemos colocado las partículas en su configuración inicial, la energía potencial está automaticamente definida y, por conservación de la energía, también la cinética. Por tanto, debemos asignar velocidades a las partículas para que se cumpla que $E_{\text{cin}} = E_{\text{tot}} - E_{\text{pot}}$.

El primer paso sería darles velocidades aleatorias a las partículas, por ejemplo entre $(-0{,}5, 0{,}5)$ para posteriormente normalizarlas a la energía cinética deseada. Para ello empleamos el siguiente código:

```
do i=1,N
```

```
2
3       vx(i)=0.5-mi_random(iduma)
4       vy(i)=0.5-mi_random(iduma)
5       vz(i)=0.5-mi_random(iduma)
6
7   end do
```

Donde, mi_random, al igual que antes devuelve un número aleatorio entre $[0, 1)$ y, por tanto, las tres velocidades podrán tomar un valor aleatorio en el intervalo deseado.

Ahora toca calcular la energía cinética de las partículas, dado que estamos en un sistema de muchos cuerpos, la energía cinética (por el teorema de König) tiene que tener en cuenta la velocidad del centro de masas:

$$E_{\text{cin}} = \sum_{i=1}^{N} \frac{1}{2} m_i v_i^2 + \frac{1}{2} M_{\text{tot}} v_{\text{cm}}^2 \tag{18}$$

Sin embargo, podemos ahorranos el término del centro de masas y calcular la energía cinética de forma usual si, y solo si, la velocidad del centro de masas es nula. Esto podemos garantizarlo si el momento lineal es nulo en todo momento. Por la segunda ley de Newton, el momento lineal se conserva si no hay fuerzas externas y dado que es el caso, si el momento inicial es nulo lo será en todos los instantes posteriores. Así que vamos a restar a la velocidad de cada partícula la suma de la velocidad por partícula haciendo así que el momento total sea nulo:

```
1   ! Debemos imponer que el momento lineal sea nulo, primer sumamos el
       momento en cada componente
2
3   sx=sum(vx)
4   sy=sum(vy)
5   sz=sum(vz)
6
7   ! Corregimos el momento lineal de cada part cula para que el momento
       lineal sea 0
8
9   vx=vx-sx/N
10  vy=vy-sy/N
11  vz=vz-sz/N
```

Y ahora normalizamos la energía cinética a aquella que se obtiene de conservación de energía, con el siguiente programa:

```
1   !   Calculamos la nueva energ a cin tica
2
3   ecin=0.5d00*(sum(vx*vx)+sum(vy*vy)+sum(vz*vz))
4
5   ! Calculamos la energ a cin tica que necesitamos para que etot sea la
       que queremos y obtenemos la relacion entre
6   ! la energ a cin tica que queremos y la que tenemos.
7
8   ecin_=etot_-epot
9   factor=dsqrt(ecin_/ecin)
10
11  !   Multiplicamos cada componente por ese factor
```

```
12
13  vx=vx*factor
14  vy=vy*factor
15  vz=vz*factor
16
17  ! Y ya tenemos la energ a cin tica
18
19  ecin=0.5d00*(sum(vx*vx)+sum(vy*vy)+sum(vz*vz))
```

5. Evolución temporal

Hasta este punto, nuestro programa ha generado, las posiciones, velocidades y aceleraciones de cada partícula, además de la energía potencial y cinética que dan lugar a la energía total que fijamos al inicio del programa. Ahora es el momento de hacer evolucionar esta configuración a un tiempo posterior. Esta evolución está gobernada, clásicamente, por la segunda ley de Newton que, claramente no es resoluble por cuadraturas, pues hay 500 partículas pero solo es conservada la energía cinética y los momentos lineal y angular (en promedio). Por tanto, deberemos resolver la ecuación numéricamente y para ello vamos a emplear el algoritmo discretizador, *Verlet*.

El esquema integrador es el siguiente:

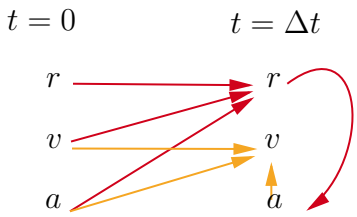

Figura 9: Esquema integrador del algoritmo *Verlet*

Es decir, el algoritmo primero partirá de una configuración inicial $r(t)$, $v(t)$ y $a(t)$ y calculará las posiciones nuevas:

$$r_i(t + \Delta t) = r_i(t) + v_i(t)\Delta t + \frac{1}{2}a_i(t)\Delta^2 t \tag{19}$$

A partir de las posiciones en $t+\Delta t$ es posible calcular la energía cinética del mismo modo que se hizo previamente, i.e, con la llamada a la subrutina pot_lj. De modo que ya tenemos $r(t+\Delta t)$ y $a(t+\Delta t)$. Para calcular $v(t + \Delta t)$ se usa las aceleraciones (remota y futura) y las velocidades remotas:

$$v_i(t + \Delta t) = v_i(t) + \frac{1}{2}[a_i(t) + a_i(t + \Delta t)]\Delta t \tag{20}$$

La manera de programar esto es mediante una subrutina que tome posiciones, velocidades, aceleraciones, energías y derivadas y nos las devuelva en un tiempo posterior,

```
1  rx=rx+vx*deltat+ax*deltat_cm
2  ry=ry+vy*deltat+ay*deltat_cm
3  rz=rz+vz*deltat+az*deltat_cm
4
5  vx=vx+deltat_m*ax
```

```
vy=vy+deltat_m*ay
vz=vz+deltat_m*az


call pot_lj(rx,ry,rz,ax,ay,az,epot,depotr,ddepotrr)

vx=vx+deltat_m*ax
vy=vy+deltat_m*ay
vz=vz+deltat_m*az

ecin=0.5d00*(sum(vx*vx)+sum(vy*vy)+sum(vz*vz))

etot=ecin+epot
```

De modo que ya hemos hecho evolucionar la configuración inicial un instante, ahora si queremos que evoluciones un tiempo concreto, por ejemplo, $t = N_{\text{pasos}}\Delta t$, con $\Delta t = 10^{-4}$ y $N_{\text{pasos}} = 5000$, debemos repetir este proceso N_{pasos} veces, por ejemplo con un bucle, como el siguiente:

```
open(10, file=fname1)

do i=1, Npasos

! Llamamos a velocity_verlet que calcula, a partir de r's, v's y a's
   antiguas, las nuevas, adem s de calcular las
! nuevas energ as, etot, ecin y epot.

    call velocity_verlet(rx,ry,rz,vx,vy,vz,ax,ay,az,ecin,epot,etot,depotr
       ,ddepotrr)

! Grabamos en un archivo cada 100 pasos, el valor del tiempo, el paso y
   las energ as.

    if (mod(i,1)==0) then

        tiempo=dble(i)*dble(deltat)
        write(10,'(e13.6, 2x,e13.6, 2x, e13.6, 2x,e13.6)') tiempo, ecin,
           etot, epot

    end if

 end do
```

Hemos añadido también un guardado de datos para ir registrando la evolución de las configuraciones, posteriormente será comentado.

6. Análisis de resultados

Al llegar a este punto ya hemos pasado de nuestra configuración inicial a otra en un tiempo t posterior. El objetivo de este programa era dejar avanzar $N_{\text{pasos}} = 5000$ para ver si había algún pequeño salto en la energía debido a la alta simetría de la configuración inicial. Obviamente para poder estudiar

esto no nos sirve con saber la configuración inicial y la final, por eso hemos ido guardando todas las configuraciones intermedias. El formato concreto ha sido e13.6, que significa que tendrá una longitud total de 13 caracteres, con 6 cifras para decimales y el resto de caracteres no ocupados para el exponente.

Y los datos de la configuración final, juntos con algún otro dato fundamental para la simulación, los hemos guardado en un archivo binario para no perder información al volverlo a usar y, para una mejor visulización, también en ASCII. Para ello hemos empleado este programa:

```
open(20, file=fname2, form='unformatted')

    write(20) N, L, V, rc, ecin, etot, epot, rx, ry, rz, vx, vy, vz, ax,
       ay, az
    write(20) Npasos, deltat, ecorr, depotr_corr, ddepotrr_corr

close(20)

open(30, file=fname3)

    write(30, '(I4,2x,I7)') N, Npasos
    write(30,'(9(2x,e13.6))') rx, ry, rz, vx, vy, vz, ax, ay, az
    write(30,'(7(2x,e13.6))') L, V, rc, ecorr, depotr_corr, ddepotrr_corr
       , deltat
    write(30,'(5(2x,e13.6))') ecin, etot, epot, depotr, ddepotrr

close(30)
```

Y finalmente representamos gráficamente la evolución temporal de las energías frente al tiempo, en las siguientes figuras se reprsentan esos resultados.

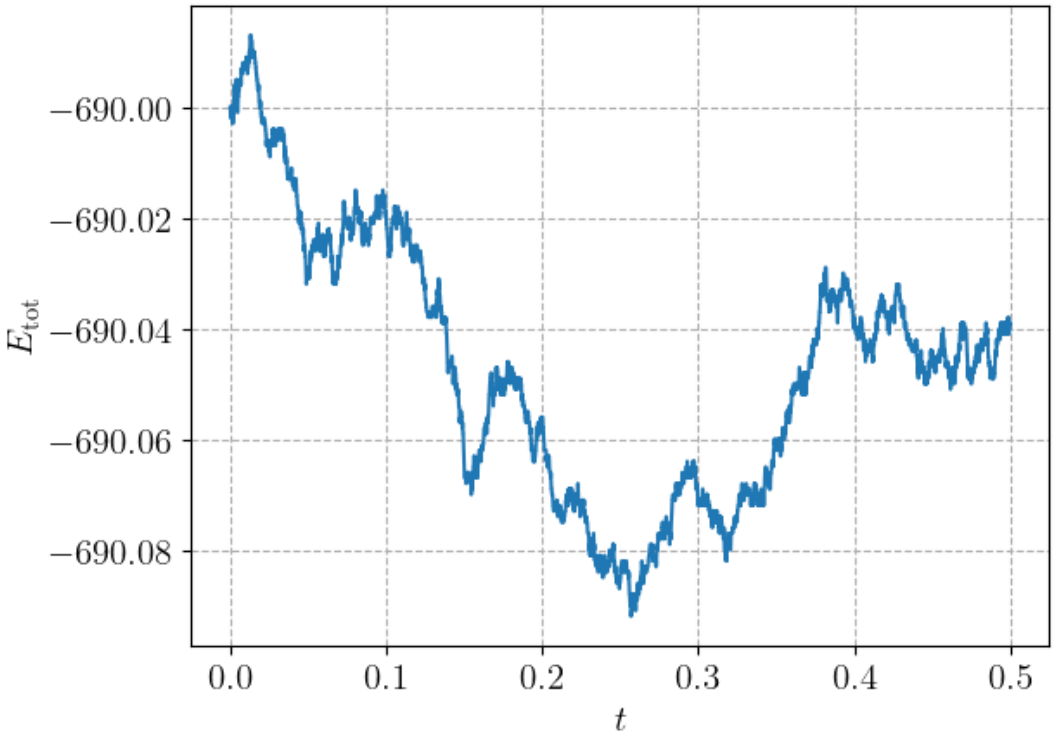

Figura 10: Energía total frente a tiempo

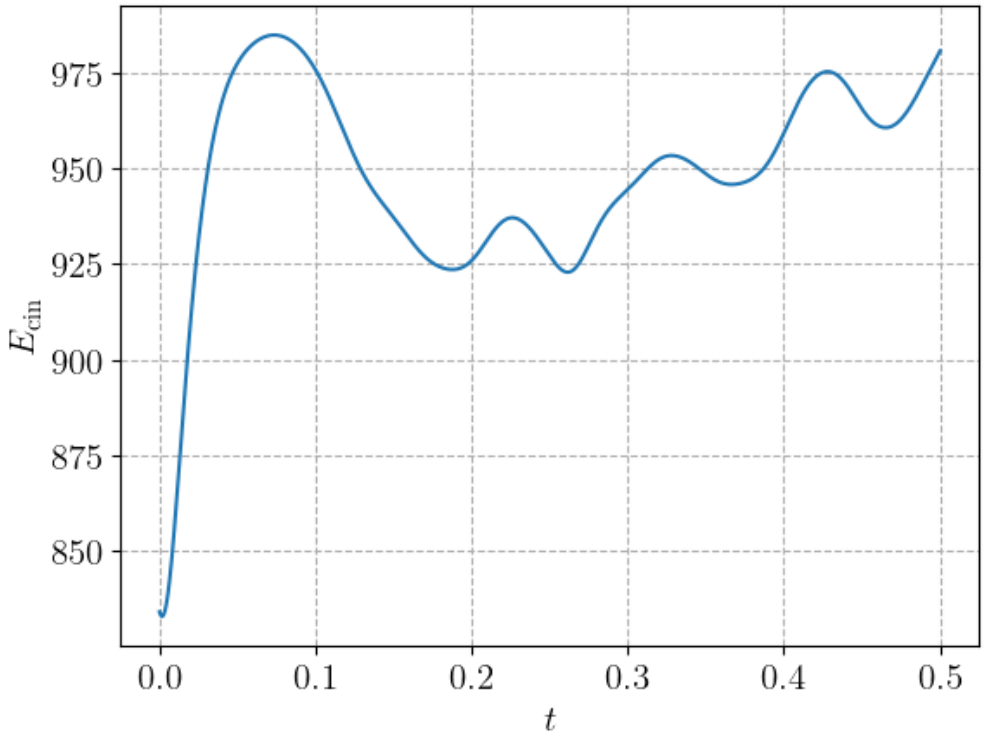

Figura 11: Energía cinética frente a tiempo

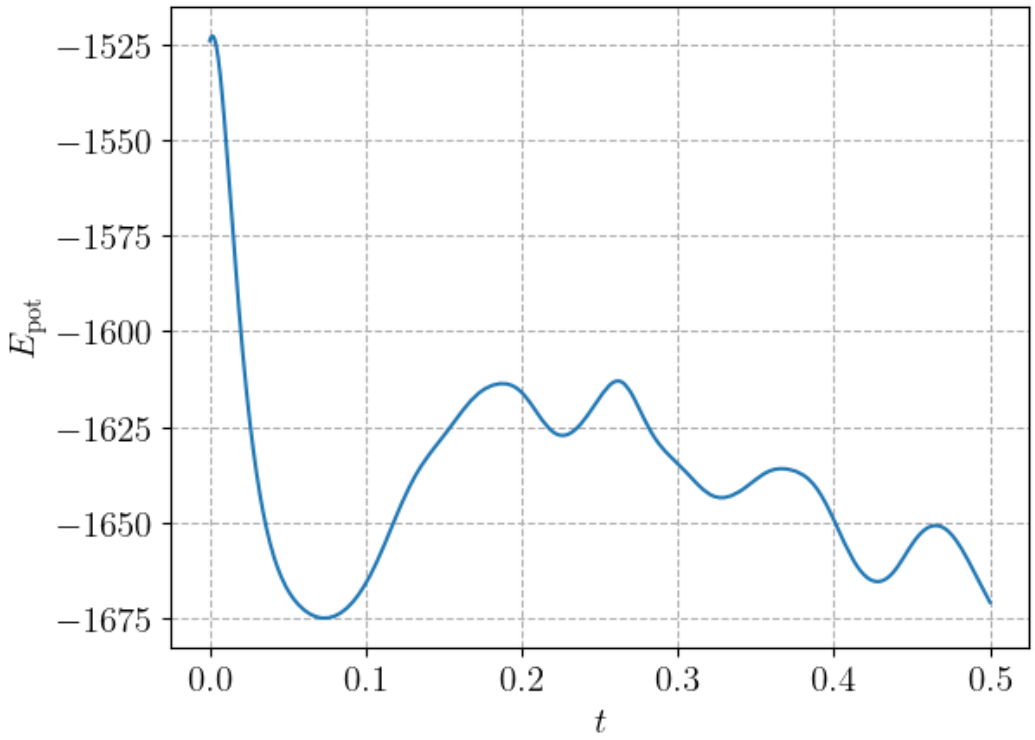

Figura 12: Energía potencial frente a tiempo

Un breve análisis nos permite concluir que la energía se conserva mejor que un 1‰ y por tanto el método de descolocación de partículas ha dado un buen resultado a la hora de evitar el salto de energías.