

Comparación entre Dinámica Molecular, "Dinámica" de Monte Carlo y Dinámica Browniana de un fluido de Lennard-Jones

Alumno: Méndez Martín

Docentes: Dra. Marconi Verónica I.; Dr. Banchio Adolfo

Universidad Nacional de Córdoba (UNC)

Facultad de Matemática, Astronomía, Física y Computación (FaMAF)

Curso de Física Computacional: Informe final

(*) Para mayor claridad se recomienda ver figuras en la versión digital.

I. RESUMEN

En el presente trabajo se busca reproducir los resultados del artículo de los autores Jean-Pierre Hansen y Loup Verlet titulado "*Phase Transitions of the Lennard-Jones System*". Para ello, se desarrollaron tres programas computacionales de un sistema de partículas bajo un potencial de interacción de Lennard-Jones y se evolucionaron los sistemas utilizando "dinámica" de Monte Carlo, dinámica molecular y dinámica Browniana, de tal forma de comparar las tres dinámicas entre sí y determinar si es posible reproducir los resultados del artículo mencionado. Para ello se describirán de forma breve las características de cada método, sus problemáticas y algunos algoritmos empleados. Luego se muestran los resultados obtenidos y detalles específicos de las simulaciones, junto con las conclusiones.

II. INTRODUCCIÓN

II-1. "Dinámica" de Monte Carlo

El método de Monte Carlo es un método de dinámica configuracional, por ello, su dinámica no es realista ya que la dinámica depende fuertemente de la aleatoriedad de los generadores random utilizados, sin imponer ninguna ecuación de evolución física. Además, el método de MC está diseñado sólo para sistemas en el equilibrio termodinámico, es decir que si bien la dinámica es artificial los valores de equilibrio (de algún observable físico) son reales.

■ ¿Cómo explorar el espacio de configuraciones?

Para explorar el espacio de configuraciones de forma ordenada, no hay dudas de que debemos consultarle a cada una de las partículas si es posible evolucionar su posición o no, y de esta forma nos aseguramos fielmente que estamos recorriendo todo el espacio de configuraciones y en consecuencia realizamos un MC_{step} . Ahora bien, al momento de explorar todo el espacio de configuraciones de forma random no estamos seguros de que tirando $(n_p \cdot n_p)$ índices visitaremos todas las partículas, pues podría ocurrir que se repitieran ciertos índices, sin embargo, si tiramos más que $(n_p \cdot n_p)$ índices, con toda seguridad, visitaremos una dada partícula dos veces y esto ya no se correspondería con un MC_{step} . Por ello, al explorar el espacio de forma aleatoria se tiran, al igual que en el caso ordenado, tantos índices como partículas tengamos. La diferencia sustancial entre estos dos métodos (ambos igualmente válidos) es que al momento de aumentar el número de partículas en el sistema simulado, el método

de exploración aleatoria tiene mejor performance (medido en tiempo de CPU) que el método de exploración ordenada.

■ Método de metrópolis

Teniendo en cuenta que el método de metrópolis es, en esencia, un caso particular de *importance sampling* donde evaluamos la energía del sistema de partículas desplazando sólo una de ellas. Por un lado, si esta nueva energía es menor que la energía de la configuración anterior se acepta la nueva posición de la partícula y el sistema evoluciona, por otro lado, si la nueva energía es mayor a la energía de la configuración anterior se acepta bajo el criterio de Boltzmann que consiste en aceptar el desplazamiento bajo la hipótesis según la ecuación 1, donde el rnd es un número random de distribución uniforme que varía de cero a uno (en el presente trabajo se trabajó con el generador de números random Mersenne Twister).

$$\exp(-\Delta U/k_B T) \geq rnd \quad (1)$$

El pseudocódigo 1 que se muestra a continuación realiza el cálculo anterior, donde se tuvieron en cuenta, en caso de no aceptar el desplazamiento, la actualización de los vectores posiciones de las partículas y en caso de aceptar los desplazamiento además las posiciones corregidas según PBC.

Listing 1: Subrutina: Método de Metrópolis

```
1 ! EVOLUCION CONFIGURACIONAL DEL SISTEMA
2 subroutine evolution_monte_carlo(n_p,x_vector,&
3 y_vector,z_vector,x_vector_noPBC,y_vector_noPBC,&
4 z_vector_noPBC,U_adim,T_adim,r_cutoff,density,&
5 delta_x,delta_y,delta_z)
6 ! Ingresamos
7 n_p ! cantidad de particulas
8 ! vectores posicion
9 x_vector(n_p),y_vector(n_p),z_vector(n_p)
10 ! componentes del vector posici n sin PBC
11 x_vector_noPBC(n_p),y_vector_noPBC(n_p),&
12 z_vector_noPBC(n_p)
13 U_adim! energia potencial adimensional
14 T_adim! temperatura adimensional
15 r_cutoff! distancia de truncado
16 density! densidad de particulas
17 ! desplazamientos optimizados
18 delta_x,delta_y,delta_z
19 ! hacemos un paso de Monte Carlo (MC_step)
20 do MC_index=1,n_p
21 ! generamos un indice de partícula rnd [0:n_p]
22 ! guardamos posicion en tiempo actual
23 ! evolucionamos posiciones con y sin PBC
24 ! correccion de posiciones segun PBC
25 ! calculamos variacion de energia potencial
26 if (deltaU_adim<=0._dp) then
27 ! modificamos energia potencial
28 U_adim=U_adim+deltaU_adim
```

```

29      ! aceptamos el cambio y salimos
30  else
31      if (T_adim==0._dp) then
32          ! modificamos vectores posiciones
33          ! segun datos originales
34          ! no aceptamos el cambio y salimos
35      end if
36      ! generamos numero random
37      ! uniforme entre [0:1)
38      nrnd=real(grnd(),dp)
39      ! tenemos en cuenta posible
40      ! overflow del argumento
41      ! de la exponencial
42      if ((deltaU_adim*(1._dp/T_adim))&
43          <abs(log(tiny(1._dp)))) then
44          ! aplicamos metodo de metropolis
45          ! seg n probabilidad de Boltzmann
46          if (exp(-deltaU_adim*(1._dp/T_adim))&
47              >=nrnd) then
48              ! modificamos energia potencial
49              ! aceptamos el cambio y salimos
50          else
51              ! modificamos vectores posiciones
52              ! segun datos originales
53              ! no aceptamos el cambio y salimos
54          end if
55      else
56          if (nrnd==0._dp) then
57              ! modificamos energia potencial
58              ! aceptamos el cambio y salimos
59          end if
60          ! modificamos vectores posiciones
61          ! segundatos originales
62          ! no aceptamos el cambio y salimos
63      end if;end if;end do
64  end subroutine evolution_monte_carlo

```

■ Propuesta de desplazamientos

Para generar nuevas configuraciones se propone un camino que gobernará la evolución de las configuraciones (coordinadas de las partículas), entonces calculamos la energía de esta nueva configuración y podremos aplicar el método de Metrópolis. En el MCD se proponen actualizar las coordinadas de una única partícula por vez (en lugar de actualizar las coordinadas de todas las partículas a la vez) de la forma $x_{new} = x_{old} + \Delta x(\text{rnd} - 0,5)$ donde rnd es un número random uniforme de 0 a 1 y se resta 0,5 para obtener un número random uniforme entre -0,5 y 0,5 asegurando estadísticamente la simetría de los desplazamientos (propiedad isotrópica). Por otro lado Δx es un paso de desplazamiento propuesto, este valor suele ser crítico en el comportamiento de las simulaciones. Si por un lado, se tiene un valor muy pequeño las nuevas configuraciones inducen cambios de energía muy pequeños y se aceptan todos los cambios de desplazamiento (pero recorro el espacio de configuraciones lentamente), si por otro lado, se tiene un valor muy grande las nuevas configuraciones inducen cambios de energía muy grande y no se acepta ningún cambio de desplazamiento (pero recorro el espacio de configuraciones rápidamente). Sin embargo, debe existir un rango entre estos dos casos en que la probabilidad de aceptación es considerable y el Δx no es tan pequeño, y donde uno esperaría ser más eficiente explorando el espacio de configuraciones. Según la literatura podemos decir que como primera aproximación un valor de Δx óptimo es tal que la probabilidad de aceptación sea de un 50 %, sin embargo, estudios más detallados sugieren que aquellos pasos de desplazamiento que permitan sólo un 20 % de probabilidad de aceptación originan simulaciones más eficientes. Con lo mencionado anteriormente, se implementó una subrutina que permite encontrar estos máximos desplazamientos tal que produzcan una probabilidad de aceptación entorno al 50 %.

En el programa 2 se muestra el fragmento de código de la subrutina mencionada.

Listing 2: Subrutine: ajuste de desplazamiento máximo (optimizado)

```

1  subroutine max_displacement_adjusting(n_p,x_vector ,
2      y_vector,z_vector,&
3      T_adim,r_cutoff,density,delta_x,delta_y,delta_z)
4      ! cantidad de particulas
5      integer(sp),intent(in)::n_p
6      ! vectores posicion
7      real(dp),intent(inout):: x_vector(n_p),&
8      y_vector(n_p),z_vector(n_p)
9      ! Temperatura de equilibrio
10     real(dp),intent(in):: T_adim
11     real(dp),intent(in):: r_cutoff! distancia de
12     truncado
13     real(dp),intent(in):: density
14     real(dp),intent(inout):: delta_x,delta_y,delta_z
15     real(dp)::deltaU_adim! variacion de energia
16     ! posiciones sin desplazar con PBC
17     real(dp):: x_old,y_old,z_old
18     real(dp):: L! desplazamientos
19     real(dp):: nrnd! numero random
20     integer(sp):: seed,seed_val(8),MC_index,index
21     integer(sp):: counter! contador de aceptaciones
22     real(dp):: accept_prob! acceptance_probability
23     ! variables para definir cota adimisible
24     real(dp),parameter:: value_prob=0.5_dp,
25     epsilon_prob=0.05_dp
26     logical:: end_loop
27     real(dp):: U_adim_old,U_adim_new
28     ! +++ DESCOMENTAR SI SE QUIERE COMPARAR CON EL
29     METODO ORIGINAL +++
30     ! U_adim_old=u_lj_total(n_p,x_vector,y_vector,
31     z_vector,r_cutoff,density)
32     L=(real(n_p,dp)*(1._dp/density))*((1._dp/3._dp)
33     call date_and_time(values=seed_val)
34     seed=seed_val(8)*seed_val(7)*seed_val(6)+
35     seed_val(5);call sgrnd(seed)
36     ! seteamos los desplazamientos a valores
37     iniciales
38     delta_x=1._dp;delta_y=1.1_dp;delta_z=1.2_dp
39     end_loop=.false.
40     do while (end_loop.eqv..false.)
41         counter=0_sp
42         ! hago un paso de Monte Carlo (MC_step)
43         do MC_index=1,n_p
44             ! random integer from 1 to n_p (elijo
45             particulas al azar y desplazo)
46             nrnd=real(grnd(),dp);index=1_sp+floor(n_p*
47             nrnd,sp)
48             ! guardamos posicion en tiempo actual
49             x_old=x_vector(index);y_old=y_vector(index);
50             z_old=z_vector(index)
51             ! evolucionamos posiciones con y sin PBC
52             nrnd=real(grnd(),dp);x_vector(index)=x_old+(
53             nrnd-0.5_dp)*delta_x
54             nrnd=real(grnd(),dp);y_vector(index)=y_old+(
55             nrnd-0.5_dp)*delta_y
56             nrnd=real(grnd(),dp);z_vector(index)=z_old+(
57             nrnd-0.5_dp)*delta_z
58             ! correccion de posiciones seg n PBC
59             call position_correction(n_p,density,x_vector(
60             index),y_vector(index),z_vector(index))
61             ! variacion de energia interna
62             deltaU_adim=delta_u_lj(n_p,x_vector,y_vector,
63             z_vector,r_cutoff,density,x_old,y_old,
64             z_old,index)
65             ! +++ DESCOMENTAR SI SE QUIERE COMPARAR CON EL
66             METODO ORIGINAL +++
67             U_adim_new=u_lj_total(n_p,x_vector,y_vector,
68             z_vector,r_cutoff,density)
69             ! deltaU_adim=U_adim_new-U_adim_old
70             cond1: if (deltaU_adim<=0._dp) then;counter=
71             counter+1_sp;exit cond1
72         else
73             if (T_adim==0._dp) then
74                 exit cond1! no aceptamos el cambio y
75                 salimos
76             end if
77             nrnd=real(grnd(),dp)

```

```

58     if ((deltaU_adim*(1._dp/T_adim))<abs(log(tiny
59         (1._dp)))) then
60         ! metodo de metropolis
61         if (exp(-deltaU_adim*(1._dp/T_adim))>=nrand)
62             then
63                 counter=counter+1_sp
64                 exit cond1 ! aceptamos el cambio y salimos
65             else
66                 exit cond1 ! no aceptamos el cambio y
67                 salimos
68             end if
69         else if
70             if (nrand==0._dp) then; counter=counter+1_sp;
71             exit cond1; end if
72         end if
73         x_vector(index)=x_old; y_vector(index)=y_old;
74         z_vector(index)=z_old
75     end do
76     ! evaluamos segun probabilidad de aceptacion y
77     ajustamos desplazamiento
78     accept_prob=real(counter,dp)/real(n_p,dp)
79     cond2: if ((value_prob-epsilon_prob)<accept_prob
80         .and. accept_prob<(value_prob+epsilon_prob)
81         ) then
82         end_loop=.true.; exit cond2
83     else
84         if (accept_prob>(value_prob+epsilon_prob)) then
85             delta_x=delta_x*1.05_dp; delta_y=delta_y*1.05
86             _dp; delta_z=delta_z*1.05_dp
87             exit cond2
88         else; delta_x=delta_x*0.95_dp; delta_y=delta_y
89             *0.95_dp; delta_z=delta_z*0.95_dp
90         end if
91     end if cond2
92 end do
93 end subroutine max_displacement_adjusting

```

■ Evitando problemas numéricos de underflows

En el método de metrópolis, para explorar el espacio de configuraciones de forma aleatoria, fue necesario obtener números random enteros, para ello, se utilizó tal como se ilustra en programa 3 (haciendo uso de la función *floor(x)*).

Listing 3: Program: generación de números random enteros

```

1 ! make clean
2 ! make test_integer_rnd.o && ./test_integer_rnd.o
3 program test_integer_rnd
4     use module_precision; use module_random_generator
5     implicit none
6     integer(sp), parameter :: m=1,n=10
7     integer(sp) :: seed, seed_val(8)
8     integer(sp) :: i,j,i_rnd,j_rnd
9     real(dp) :: nrand
10    ! generate seed
11    call date_and_time(values=seed_val)
12    seed=seed_val(8)*seed_val(7)*seed_val(6)+&
13        seed_val(5)
14    ! floor(x,type) returns the greatest integer
15    ! less than or equal to x
16    do i=1,n; do j=1,n
17        ! We want to choose one from m to n integers
18        nrand=ran2(seed)
19        i_rnd=m+floor((n+1-m)*nrand,sp) ! rows
20        nrand=ran2(seed)
21        j_rnd=m+floor((n+1-m)*nrand,sp) ! columns
22        write(*,*) i_rnd,j_rnd
23    end do; end do
24 end program test_integer_rnd

```

Por otra parte, como en el algoritmo de metrópolis es necesario computar la función exponencial decreciente de argumentos muy grandes, se corre el riesgo de producir *underflows* de valores, por ello, se utilizó la función *tiny(x)* tal como se muestra en el programa 4.

Listing 4: Program: Solución a underflows en función exponencial enteros

```

1 ! make clean && make test_underflows_exp_function.o
2 ! ./test_underflows_exp_function.o
3 program test_underflows_exp_function
4     use module_precision
5     implicit none
6     real(dp) :: x,result
7     integer(sp) :: i
8     ! tiny(x) returns the smallest positive
9     ! (non zero) number in the model of the type of
10    x
11    write(*,*) 'abs(log(tiny(x)))=',&
12        abs(log(tiny(1._dp)))
13    do i=1,800
14        x=real(i,dp)
15        if (x<abs(log(tiny(result)))) then
16            result=exp(-x)
17        else; result=0._dp; end if
18        write(*,*) x,result
19    end do
20 end program test_underflows_exp_function
21 ! Note: https://gcc.gnu.org/onlinedocs/gcc-4.5.4/
22 gfortran/TINY.html

```

■ Evitando el quenching

En todas las simulaciones de MCD se llevo al sistema al equilibrio de forma gradual hasta llegar a la temperatura de referencia propuesta, es decir, partiendo de la configuración inicial del sistema de partículas, este se evoluciono un paso de Monte Carlo desde una temperatura inicial con valor igual al 10 % de la temperatura de referencia hasta una temperatura con un valor igual al 95 % de la temperatura de referencia. Esto nos permite evolucionar el sistema con deltas términos no tan abruptos (evitando el **quenching** y en consecuencia cada paso de temperatura comienza con una configuración inicial de posiciones de partículas según evolucionó el sistema a la temperatura inmediatamente anterior, de esta forma se reduce el tiempo de computo y se simula de acuerdo a parámetros más realistas (según los experimentos).

II-2. Dinámica Molecular

La simulación de tipo dinámica molecular (DM) es una técnica útil en cálculos de equilibrio y propiedades de transporte de sistemas clásicos de muchos cuerpos. Y se trata de sistemas clásicos debido a que los centros de masa de las partículas constituyentes del sistema se mueven de acuerdo las leyes de la mecánica clásica y donde se estudian comportamientos en rangos donde los efectos cuánticos son despreciables. Por otro lado, las simulaciones de dinámica molecular comparten muchas propiedades de los experimentos reales, es decir, primeramente preparamos la muestra, aplicamos un sistema modelo consistente de n_p partículas y resolvemos las ecuaciones clásicas de movimiento (mecánica Newtoniana, Lagrangiana o Hamiltoniana) para este sistema hasta que las propiedades del sistema no cambien más con el tiempo (sistema en equilibrio), luego del equilibrio realizamos las mediciones de las propiedades correspondiente y para ello debemos expresar los observables que mediremos en función de las coordenadas generalizadas del sistema (posiciones y momentos).

■ Temperatura y corrección de velocidades

El significado físico clásico de la temperatura es que es una cantidad estadística y usando el teorema de equipartición de la energía, que involucra a todos los grados de libertad que aparecen como términos cuadráticos en el hamiltoniano del sistema, podremos expresar la energía cinética promedio por grado de libertad de la siguiente manera,

$$\left\langle \frac{1}{2} m(v_\alpha)^2 \right\rangle = \frac{1}{2} k_B T \quad (2)$$

ahora bien, como la energía cinética total del sistema fluctua podemos definir la temperatura instantánea como muestra la ecuación 3.

$$T(t) = \sum_{i=1}^{n_p} \frac{m_i [v_i(t)]^2}{k_B N_f}; N_f = 3n_p \quad (3)$$

donde N_f se refiere a los grados de libertad del sistema.

Por otro lado, debido al numero finito de partículas la temperatura puede modificarse demasiado debido al comportamiento de ciertas partículas en la simulación que al tener un sistema finito es relevante. Por ello, tanto en la termalización como en el régimen estacionario es necesario asegurarse de que no se produzcan cambios muy grandes de la temperatura respecto a la de referencia, para lograr esto, se produce un re-escalado de las velocidades en todos los pasos utilizando el *termostato* más usual que se muestra en la ecuación 4.

$$\vec{v}(t_{new}) = \lambda \vec{v}(t_{old}); \lambda = \sqrt{\frac{T_{ref}}{T(t_{new})}} \quad (4)$$

De esta forma logramos reproducir aproximadamente el ensemble canónico para realizar nuestros promedios de observables parámetros constantes de número de partículas n_p , temperatura adimensional de referencia T_{adim} y volumen del sistema L^3 .

■ Configuración inicial de parámetros

Teniendo en cuenta que las simulaciones de DM se encargan de muestrear todo el espacio de configuraciones, el cual su éxito en dicha exploración dependerá únicamente del tiempo que le demos para recorrer el espacio y no de las configuraciones iniciales dadas, por ello, si bien sabemos que idealmente en el equilibrio, las componentes de la velocidad seguirán una distribución Gaussiana y la velocidad total seguirá una distribución de Maxwell-Boltzmann, para reducir tiempo de simulación posiblemente se deba invertir un esfuerzo en proponer estas distribuciones de velocidad para el estado inicial, sin embargo, la ganancia de tiempo de CPU en las simulaciones de DM están fuertemente influenciadas por el cuello de botella del algoritmo, que es, el cálculo de las fuerzas de interacción entre pares de partículas. Entonces, debido a que no se gana demasiado en proponer distribuciones específicas en las velocidades iniciales, se propuso una distribución random uniforme en el rango $[-0.5; 0.5]$ para cada componente de la velocidad (utilizando el generador de números pseudoaleatorios Mersenne Twister y corrigiendo escalando dichas velocidades según el termostato descrito en II-2 para que las partículas cuenten con temperatura (energía cinética) realista), por otro lado, cuando nos referimos a las velocidades de las partículas, las medimos respecto del centro de masas del sistema, por tanto, la presión y la temperatura del sistema de partículas no se modifican si el recipiente que lo contiene está en movimiento, para ello a las velocidades aleatorias se le resta la velocidad del centro de masas. Luego, se escalan las velocidades según el termostato descrito en la sección II-2. Cabe mencionar que las posiciones iniciales según la estructura FCC podrían modificarse restándoles las distancias que recorrería cada partícula con las velocidades iniciales definidas anteriormente y luego corregir dichas posiciones según PBC, sin embargo, esta consideración es irrelevante respecto a los resultados obtenidos en el equilibrio e introduce cierto tiempo de CPU que desea reducirse, por ello no fueron consideradas en las simulaciones y se partió de una configuración posicional de las partículas según la estructura FCC.

■ Integración de las ecuaciones de movimiento

Teniendo en cuenta que las simulaciones de Dinámica Molecular, a diferencia de las simulaciones configuracionales de Monte Carlo, siguen una dinámica realista, por ello, las ecuaciones de movimiento deben cumplir con leyes de conservación de momento lineal, momento angular y la energía total del sistema (para sistemas con vínculos holónomos e independientes del tiempo, y cuyo potencial no depende de las velocidades entonces, el hamiltoniano es igual a la energía del sistema y si además el sistema es invariante ante traslaciones temporales el hamiltoniano, y en consecuencia la energía, se conserva). Las PBC inducen pérdida de simetría rotacional del sistema y en consecuencia el momento angular no puede conservarse en simulaciones que utilicen PBC. Los métodos de integración más utilizados son algoritmo de Verlet clásico, algoritmo de Leap-Frog, los cuales son inestables (estrictamente el algoritmo de Verlet clásico no conserva la energía del sistema sino que conserva la energía de un pseudo-hamiltoniano que en el límite de tiempos cortos es igual al hamiltoniano real), y el algoritmo de velocity-Verlet, que es estable. Este último es el que aplicamos en el presente trabajo. Cabe aclarar que se aplicó una variante del algoritmo que no requiere mantener guardado el vector fuerza en el tiempo anterior para computar las velocidades finales, debido a que se integra en dos pasos las ecuaciones de movimiento, utilizando un paso intermedio ficticio para aproximar las soluciones y disminuyendo el espacio en memoria de las simulaciones por cada llamada al integrador. La subrutina con el fragmento de código que resuelve la integración se muestra en 5.

Listing 5: Subroutine: Integración de las ec. de movimiento

```

1 subroutine velocity_verlet(n_p, x_vector, y_vector,
2   z_vector, &
3   x_vector_noPBC, y_vector_noPBC, z_vector_noPBC, &
4   vx_vector, vy_vector, vz_vector, &
5   delta_time, mass, r_cutoff, density, force_x, force_y,
6   force_z)
7   integer(sp), intent(in) :: n_p
8   real(dp), intent(inout) :: x_vector(n_p), y_vector(n_p),
9   z_vector(n_p)
10  real(dp), intent(inout) :: x_vector_noPBC(n_p),
11  y_vector_noPBC(n_p), &
12  z_vector_noPBC(n_p) ! componentes del vector
13  posici n sin PBC
14  real(dp), intent(inout) :: vx_vector(n_p), vy_vector(n_p),
15  vz_vector(n_p)
16  real(dp), intent(in) :: delta_time, mass, r_cutoff,
17  density
18  real(dp), intent(inout) :: force_x(n_p), force_y(n_p),
19  force_z(n_p)
20  integer(sp) :: i
21  real(dp) :: factor
22  factor = delta_time * 0.5_dp * (1._dp / mass)
23  do i = 1, n_p
24    ! COMPONENTES DE LA VELOCIDAD A TIEMPO SEMI-
25    EVOLUCIONADO
26    vx_vector(i) = vx_vector(i) + force_x(i) * factor
27    vy_vector(i) = vy_vector(i) + force_y(i) * factor
28    vz_vector(i) = vz_vector(i) + force_z(i) * factor
29    ! COMPONENTES DE LA POSICION A TIEMPO EVOLUCIONADO
30    (con y sin pBC)
31    x_vector(i) = x_vector(i) + vx_vector(i) * delta_time
32    x_vector_noPBC(i) = x_vector_noPBC(i) + vx_vector(i) *
33    delta_time
34    y_vector(i) = y_vector(i) + vy_vector(i) * delta_time
35    y_vector_noPBC(i) = y_vector_noPBC(i) + vy_vector(i) *
36    delta_time
37    z_vector(i) = z_vector(i) + vz_vector(i) * delta_time
38    z_vector_noPBC(i) = z_vector_noPBC(i) + vz_vector(i) *
39    delta_time
40    call position_correction(n_p, density, &
41    x_vector(i), y_vector(i), z_vector(i))
42  end do

```

```

30 ! ACTUALIZAMOS COMPONENTES DE LA FUERZA A TIEMPO
    EVOLUCIONADO
31 call f_lj_total(x_vector,y_vector,z_vector,r_cutoff
    ,n_p,density,&
32 force_x,force_y,force_z)
33 ! COMPONENTES DE LA VELOCIDAD A TIEMPO EVOLUCIONADO
34 do i=1,n_p
35 vx_vector(i)=vx_vector(i)+force_x(i)*factor
36 vy_vector(i)=vy_vector(i)+force_y(i)*factor
37 vz_vector(i)=vz_vector(i)+force_z(i)*factor
38 end do
39 end subroutine velocity_verlet

```

■ Más sobre estabilidad e inestabilidad

Obtener precisión para tiempos largos es muy importante, porque cuanto más largo sea el paso temporal que podemos utilizar en los integradores, menos evaluaciones de las fuerzas se necesitan por unidad de tiempo de simulación. Por lo tanto, esto sugeriría que es ventajoso utilizar un algoritmo de integración que permita el uso de un paso de tiempo largo. Los algoritmos que permiten el uso de un paso temporal largo lo consiguen almacenando información sobre derivadas de orden cada vez más alto de las coordenadas de las partículas. En consecuencia, tienden a requerir más almacenamiento de memoria. Por otro lado, la conservación de la energía es un criterio importante (más aún si queremos realizar una simulación en el ensamble microcanónico), sin embargo, deberíamos distinguir dos tipos de conservación de la energía, aquella a tiempos cortos y aquella a tiempos largos. Los algoritmos de orden superior tienden a tener una muy buena conservación de la energía para tiempos cortos. Sin embargo, suelen tener la característica indeseable de que la energía global se desplaza durante tiempos largos. Por el contrario, los algoritmos de tipo Verlet tienden a tener una conservación de la energía moderada a corto plazo, pero poca deriva a largo plazo. Es más, para la mayoría de sistemas la trayectoria del sistema a través del espacio de fase depende sensiblemente de las condiciones iniciales. Esto significa que dos trayectorias que inicialmente están muy cerca divergen exponencialmente a medida que avanza el tiempo. Podemos considerar el error de integración provocado por el algoritmo como la fuente de esta pequeña diferencia inicial, entre la trayectoria real del sistema y la generada en nuestra simulación. Es de esperar que cualquier error de integración, por pequeño que sea, siempre ocasione que nuestra trayectoria simulada diverja exponencialmente de la trayectoria verdadera compatible con las mismas condiciones iniciales. Esto se denomina inestabilidad de Lyapunov. Aunque hay que aclarar que el objetivo en una simulación de DM no es predecir con exactitud lo que le ocurrirá a un sistema que ha sido preparado en un estado inicial conocido con precisión ya que siempre nos interesan las predicciones estadísticas.

II-3. Dinámica Browniana

El movimiento *browniano* (BD) se podría definir como el movimiento de partículas (soluto de tamaño mesoscópico) inmerso en un líquido (solvente de tamaño microscópico), el cual se caracteriza por ser un movimiento azaroso. Ejemplos de estos movimientos podrían ser: el movimiento de partículas de polvo en el aire (haz de luz que emite un proyector), partículas de humo en el aire (bocanada de humo que expulsa un fumador hacia el aire) ó partículas de polen inmersas en agua.

- Fuerza hidrodinámica, fuerza estocástica y escalas de tiempo

Si una partícula mesoscópica se sumerge en un medio compuesto de partículas microscópicas y considerando a este medio microscópico como un fluido laminar, entonces, según la hidrodinámica podemos decir que la partícula mesoscópica experimentará una fuerza de fricción que dependerá de su velocidad (fuerza viscosa). Por otro lado, la partícula mesoscópica experimentará fuerzas debidas al gran número de colisiones que ocurren entre las partículas microscópicas hacia las mesoscópicas (en general del orden de $\mathcal{O}(10^{23})$), estas fuerzas varían de forma azarosa y violenta (en escalas de tiempo del orden de $\tau_s \sim \mathcal{O}(10^{-13})[s]$), pues si hacemos observaciones en las escalas de tiempo mesoscópicas ($t \gg \tau_s$) habrán ocurrido muchas colisiones y observaremos un efecto promedio de estas fuerzas estocásticas, sin embargo, en esta escala de tiempo las fuerzas viscosas comenzarán a experimentarse y variarán muy poco (ver figuras 1, 2). La fuerza estocástica

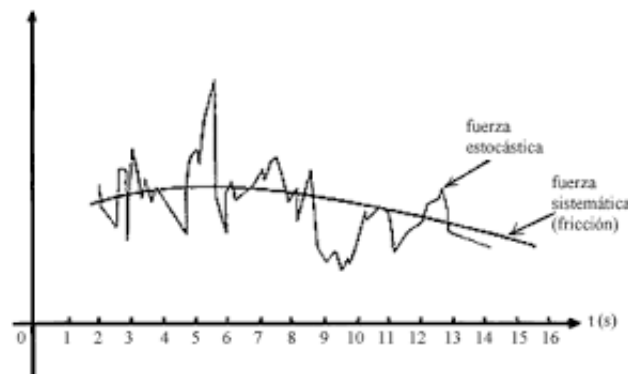


Fig. 1. Variación de fuerza estocástica y viscosa, diferencia de escalas temporales

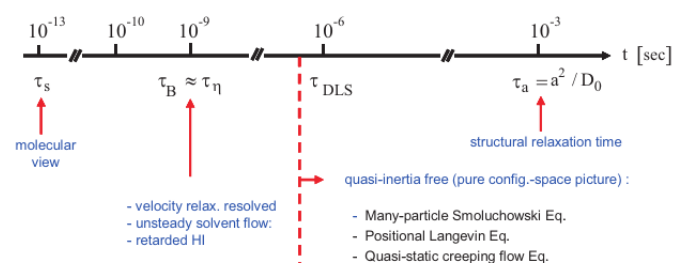


Fig. 2. Escalas temporales: valores típicos para partículas de radio de $100[nm]$ suspendidas en agua. τ_{DLS} es el tiempo de resolución para la dinámica de difracción de luz

describe entonces el efecto acumulativo de la gran cantidad de colisiones que producen las moléculas del solvente sobre el soluto. Estas colisiones, se pueden considerar estadísticamente independientes para tiempos largos comparados con τ_s y, de acuerdo al teorema central del límite al suma de muchas variables aleatorias con segundo momento finito, la distribución de probabilidad converge a una distribución gaussiana, entonces la fuerza estocástica puede ser descrita completamente como una cantidad fluctuante cuyas propiedades estadísticas están completamente determinadas por una distribución gaussiana.

■ Ecuación de velocidad de Langevin

Considerando una partícula esférica de masa m (cuyo radio es del orden de $\mathcal{O}(10^{-6}, 10^{-9})[m]$) suspendida en un solvente (en régimen de flujo laminar $Re \ll 1$) compuesto de moléculas (cuyo radio es del orden de $\mathcal{O}(10^{-10})[m]$) entonces, si observamos el movimiento de esta partícula mesoscópica con una resolución temporal que cumple $\Delta t \gg \tau_s$ entonces

la ecuación de movimiento para la velocidad traslacional $\vec{v}(t) = \dot{\vec{r}}(t)$ está dada por la ecuación de Langevin según la ecuación 5, donde el primer término del lado derecho corresponde a la fuerza de fricción promedio y el segundo del lado derecho corresponde a la fuerza estocástica fluctuante. Por otro lado, el lado izquierdo corresponde a la fuerza inercial que experimenta la partícula.

$$m \ddot{\vec{r}}(t) = -\zeta_0 \dot{\vec{r}}(t) + \vec{F}^s(t) \quad (5)$$

Cabe mencionar que, el primer término de la derecha es una aproximación de las fuerzas hidrodinámicas, válida en el límite de bajas densidades del soluto. Es decir, cuando la presencia de las otras partículas mesoscópicas puede ser ignorada. Además, en la escala temporal donde se realizan las observaciones el solvente se comporta como un medio continuo y como estamos tratando con un flujo en régimen laminar (comportamiento según flujo de Stokes, linealizando la ecuación de Navier-Stokes, a bajo número de Reynolds), por lo tanto, el coeficiente de fricción ζ_0 de una esfera puede aproximarse según la relación $\zeta_0 = 6\pi\eta_0 a$ donde η_0 es la viscosidad dinámica de el fluido y a el radio de la partícula esférica.

■ Teorema de fluctuación-disipación

La relación de fluctuación-disipación es la piedra fundamental de la teoría de respuesta lineal. Ésta relaciona la relajación de un sistema perturbado débilmente con las fluctuaciones espontáneas en equilibrio térmico. En otras palabras, si un sistema está en el instante inicial en un estado fuera del equilibrio, el sistema no sabe si fue puesto en tal estado por una fuerza externa o como resultado de una fluctuación al azar, entonces, la evolución subsecuente hacia el equilibrio será la misma en ambos casos. Por lo tanto, el teorema de fluctuación-disipación nos permite encontrar la relación que vincula las fuerzas viscosas (o de resistencia hidrodinámica) con las fuerzas estocásticas pues, el arrastre disipa energía cinética transformándola en calor y la fluctuación correspondiente es el movimiento browniano (movimiento azaroso) y este movimiento convierte energía térmica en energía cinética nuevamente.

■ Ecuación de posición de Langevin

Si realizamos observaciones en la escala temporal $t \gg \tau_B$ (donde $\tau_B \gg \tau_s$ es el tiempo de relajación de los momentos) entonces el desplazamiento cuadrático medio resulta lineal, en otras palabras, no resolvemos la relajación del movimiento y consideramos que las velocidades ya cuentan con una distribución de Maxwell-Boltzmann. Además, para observar un cambio configuracional (posicional) las partículas mesoscópicas deberán difundir en el solvente una distancia comparable con su propio tamaño, entonces, se define el tiempo de relajación estructural (el cual es necesario para cambios configuracionales apreciables) como $\tau_a = a^2/D_0$, donde $D_0 = k_B T/\eta_0$ es el coeficiente de fricción. Entonces, para tiempos $\tau_a \gg \tau_B$ los momentos ya habrán relajado completamente y sólo resolveremos la parte configuracional del problema. Una resolución temporal $\Delta t \gg \tau_B$ tiene asociada una resolución configuracional o espacial $\Delta x \gg l_B = \sqrt{D_0 \tau_B}$ y los problemas analizados en esta escala se rigen por la dinámica browniana y se estudian procesos de difusión según la dinámica de Smoluchowski donde el término inercial es despreciable (esto ocurre si las densidades del soluto y el solvente son del mismo orden, y el tiempo de respuesta es muy corto, tenemos un sistema sobreamortiguado. "Free-draining

approximation"). Teniendo en cuenta que $l_B \sim \mathcal{O}(10^{-3})[m]$ la difusión de partículas coloidales en la escala temporal de BD se verifican utilizando técnicas de difracción de luz (*dynamic light scattering techniques*).

■ Integración de las ecuaciones de movimiento

La dinámica browniana al igual que la dinámica molecular es una dinámica realista y al igual que el método de monte carlo, es una dinámica configuracional. Las ecuaciones de movimiento son las correspondiente a las ecuaciones de Langevin posicional que, integrada de forma conveniente, se puede escribir en la forma:

$$\vec{r}_i(t + \Delta t) = \vec{r}_i(t) + \frac{\vec{f}_i(t)}{3\pi\eta\sigma} \Delta t + \vec{r}_i^s(\Delta t) \quad (6)$$

donde $\vec{f}_i(t)$ es la fuerza de interacción que experimenta la partícula i debido a las otras partículas (esta fuerza es la derivada del potencial de interacción de Lennard-Jones) y $\vec{r}_i^s(\Delta t)$ es el desplazamiento browniano, el cual es una variable aleatoria de distribución gaussiana que cumple con:

$$\langle \vec{r}_i^s(\Delta t) \rangle = 0; \langle \vec{r}_i^s(\Delta t) \vec{r}_j^s(\Delta t) \rangle = 2D_0 \Delta t \delta_{ij} \delta(\Delta t) \quad (7)$$

II-4. Potencial de Lennard-Jones

En el presente trabajo consideramos como potencial de interacción el potencial interatómico de Lennard-Jones (ver figura 3) el cual sirve como modelo de fuerzas de enlace entre pares de moléculas y es utilizado para calcular la fuerza de van der Waals. Este potencial depende básicamente de tres parámetros (ver figura 8) r_{ij} que es la distancia relativa entre centros de masa de un par de partículas, ϵ que es la magnitud del pozo de potencial y σ es la distancia interatómica para la cual el potencial se anula. Además, podemos notar que el primer término del potencial tiene en cuenta qué tan intensa es la repulsión entre el par de partículas a medida que aumenta (o disminuye) la distancia interatómica, y el segundo término tiene en cuenta qué tan intensa es la atracción entre el par de partículas a medida que aumenta (o disminuye) la distancia interatómica.

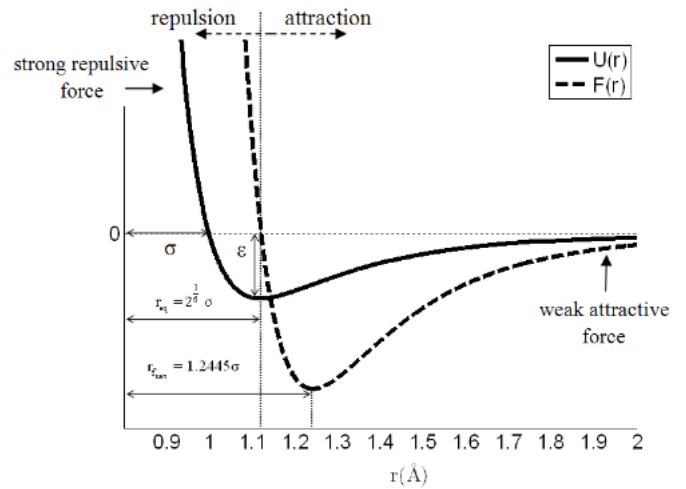


Fig. 3. Lennard-Jones: Potencial y fuerzas de interacción vs distancia interatómica

$$u_{ij} = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^6 \right] \quad (8)$$

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$$

Ahora bien, para las simulaciones es necesario adimensionalizar todas las magnitudes que tengan dimensiones, entonces el potencial individual quedará de la forma según 9.

$$u_{ij}^* = 4 \left(\frac{1}{r_{ij}^*} \right)^6 \left[\left(\frac{1}{r_{ij}^*} \right)^6 - 1 \right]; u_{ij}^* = \frac{u_{ij}}{\epsilon}; r_{ij}^* = \frac{r_{ij}}{\sigma} \quad (9)$$

Además, cabe mencionar que para evitar una discontinuidad en el potencial se utiliza el potencial truncado y desplazado (Lennard-Jones truncated shifted potential - LJTS) que se define de la siguiente manera;

$$u_{ij}^* = 4 \left(\frac{1}{r_{ij}^*} \right)^6 \left[\left(\frac{1}{r_{ij}^*} \right)^6 - 1 \right] - \dots \dots - 4 \left(\frac{1}{r_{cutoff}^*} \right)^6 \left[\left(\frac{1}{r_{cutoff}^*} \right)^6 - 1 \right] \quad (10)$$

y en la figura 4 se pueden observar las diferencias entre el método de truncado simple del potencial y el método de truncado y desplazado.

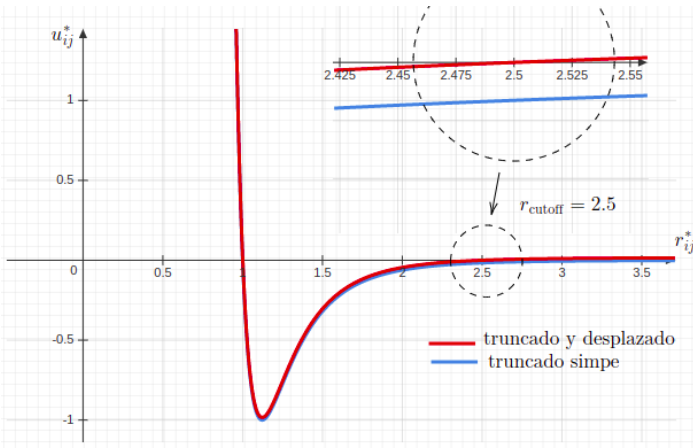


Fig. 4. LJTS vs distancia interatómica

Por otro lado, la fuerza interatómica se define como el opuesto al gradiente del potencial entre pares y en forma adimensional quedará según la ecuación 11. Y en componentes será según 12.

$$f_{r,ij}^* = -\frac{\partial u_{ij}^*}{\partial r_{ij}^*} = 24 \frac{1}{r_{ij}^*} \left(\frac{1}{r_{ij}^*} \right)^6 \left[2 \left(\frac{1}{r_{ij}^*} \right)^6 - 1 \right] \quad (11)$$

$$f_{x_k,ij}^* = \frac{r_{x_k,ij}^*}{r_{ij}^*} f_{r,ij}^*; x_k = \{x, y, z\} \quad (12)$$

Finalmente el potencial total será la suma de los potenciales individuales como muestra la ecuación 13. Notemos que se ha definido una distancia de corte entre pares r_{cutoff} este importante parámetro nos permite decidir hasta que distancia podemos considerar al potencial de interacción entre pares relevante, es decir, para distancias interatómicas mayores a este radio el potencial es despreciable y lo consideramos, a efectos prácticos, nulo, en otras palabras, este radio de corte nos permite determinar cuántas partículas vecinas interactúan cuando nos situamos en una dada partícula, la magnitud de este radio estará fuertemente influenciado respecto a si estamos tratando con interacciones de largo o corto alcance, además, deberá ser menor a la mitad de la longitud L de la celda periódica como veremos luego.

$$U_{tot} = \sum_{j=2}^{n_p} \sum_{i=1}^{j-1} u_{ij}^*; \forall r_{ij}^* \leq r_{cutoff} \quad (13)$$

II-A. Condiciones de contorno periódicas y corrección por imagen mínima

Teniendo en cuenta que se quieren estudiar propiedades en el equilibrio termodinámico ($n_p \rightarrow \infty$) y debido a que no nos interesan estudiar efectos de superficie, es decir, excluyendo el comportamiento de las partículas cerca del borde físico del sistema macroscópico, podremos eliminar dicho borde considerando una celda de estudio que se repite infinitamente, es decir, llenamos el espacio de copias idénticas de estas regiones de simulación. De esta manera una partícula que sale de esta región de estudio por alguna delimitación particular de esta, deberá ser remplazada inmediatamente por otra que tiene el mismo momento lineal desde la delimitación opuesta a la original, de esta forma actuará una condición de contorno periódica (PBC) sobre cada partícula. En la figura 5 se muestra esquemáticamente en qué consisten las PBC en un sistema partículas en 2D, en este caso cada partícula podrá cruzar la región de simulación por cualquiera de los cuatro borde, sin embargo, en 3D (como es nuestro caso) las partículas podrán cruzar la región por cualquiera de los 6 bordes de la celda tridimensional de estudio. Por el hecho de considerar PBC el

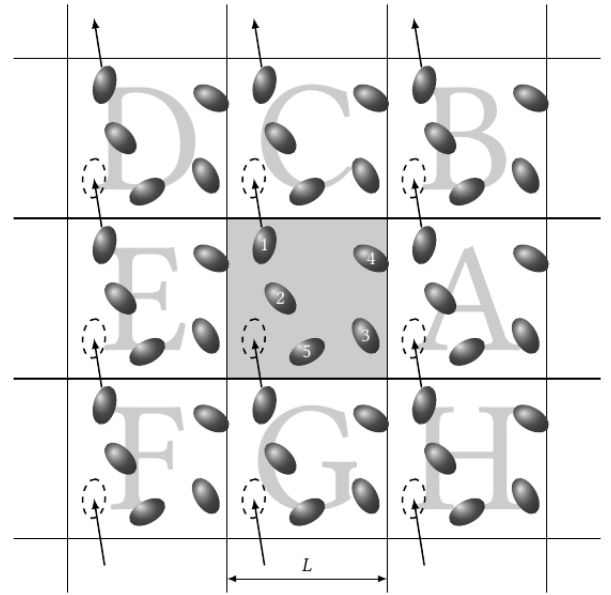


Fig. 5. Condiciones de contorno periódicas en un sistema 2D

cálculo de las fuerzas también requerirán ciertas correcciones, llamadas corrección de desplazamientos por imagen mínima las cuales consisten en que, al momentos de centrarnos en una determinada partícula para computar la fuerza neta actuando sobre ella debido a la interacción con las otras partículas del entorno, deberemos centrar también la celda de estudio en la propia partícula y considerar las distancias interatómicas dentro de esta nueva celda y, notando que las distancias relativas serán siempre menores o iguales a la mitad de la celda cúbica se deberán corregir todas aquellas distancias que superen la mitad de la celda cúbica de simulación. En la figura 6 se observa esquemáticamente esta convención para corregir el desplazamiento según imagen mínima.

III. RESULTADOS Y DISCUSIONES

Para un primer análisis se se consideró un sistema de $n_p = 256$ partículas, densidad $\rho = 0,8$ (partículas por unidad de

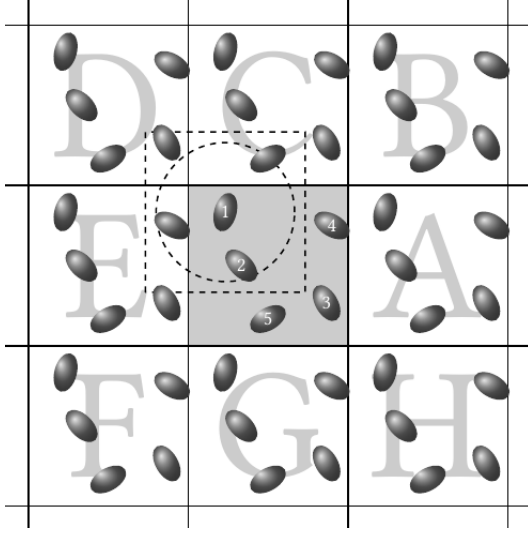


Fig. 6. Correcciones según imagen mínima en un sistema 2D

volumen), temperatura de referencia adimensional de $T_{ref} = 0,75$ y radio de corte adimensional de $r_{cutoff} = 2,5$. Para la integración de las ecuaciones de movimiento según dinámica molecular (MD) se usó un paso temporal adimensional de $\Delta t = 0,005$, mientras que para dinámica Browniana (BD) se usó un paso temporal adimensional de $\Delta t = 0,001$ y una viscosidad dinámica adimensional de $\eta_0 = 2,87$.

En la figura 18 se puede observar una animación de la dinámica Browniana y dinámica Molecular para unos parámetros específicos y en la sección IV se puede consultar el video completo acumulando un total de 100 imágenes para su implementación y usando el paquete de librerías ffmpeg.

III-A. Configuración de estructura cristalina inicial

La configuración de las posiciones iniciales de las partículas considerada fue la de una estructura cúbica centrada en las caras (FCC), la cual tiene un total de 4 átomos por celda unidad, y cuyos vectores primitivos son $\vec{a}_1 = \frac{a}{2}(\hat{e}_x + \hat{e}_y)$; $\vec{a}_2 = \frac{a}{2}(\hat{e}_y + \hat{e}_z)$; $\vec{a}_3 = \frac{a}{2}(\hat{e}_x + \hat{e}_z)$, entonces, como el total de partículas debe conservarse, la celda unidad deberá repetirse un cierto número de veces de tal forma de respetar este vínculo y además cumplir con la densidad impuesta externamente ρ . El código desarrollado corresponde a la subrutina **initial_lattice_configuration** dentro de los módulos **module_md_lennard_jones.f90**, **module_bd_lennard_jones.f90**, **module_mc_lennard_jones.f90** y en la figura 7 se puede observar la disposición de partículas en el estado inicial, la misma fue centrada en el rango $[-L/2; L/2]$.

III-B. ¿Cómo determinamos los pasos de equilibración?

Para asegurarnos de equilibrar correctamente los sistemas según la dinámica empleada, se procedió a evolucionar el sistema de partículas para valores fijos de temperatura adimensional $T_{ref} = 0,75$ y densidad reducida $\rho = 0,8$. Entonces, teniendo en cuenta que la magnitud de la energía potencial total en el equilibrio es independiente de la dinámica todas ellas deberán converger al mismo valor de energía potencial. Las gráficas correspondientes se pueden observar en la figura 13, en las cuales se graficaron las energías potenciales adimensionalizadas con el número de partículas (y con el tail correction que mencionaremos más adelante) en función de los pasos de evolución según la dinámica correspondiente. En estas gráficas se pudo observar que para el caso de DM

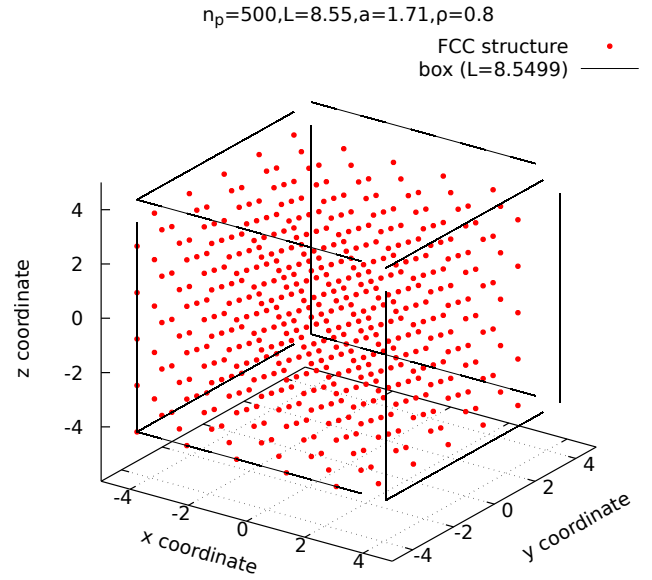


Fig. 7. Posiciones iniciales de las partículas: Estructura FCC

el sistema se equilibra para valores $MD_{step} > 3000$ pasos de evolución, para el caso de BD el sistema se equilibra para valores $BD_{step} > 80000$ pasos de evolución y para el caso de MCD el sistema se equilibra para valores $MCD_{step} > 6000$ pasos de evolución.

III-C. Estado transitorio y estacionario

Con el análisis mencionado anteriormente, para llevar a cabo la equilibración del sistema (estado transitorio) se implementaron las siguientes cantidad de pasos según la dinámica considerada:

- En dinámica molecular, para el régimen transitorio $MD_{step} = 5000$ pasos de integración y para el régimen estacionario $MD_{step} = 1000$ pasos de integración.
- En dinámica Browniana, para el régimen transitorio $BD_{step} = 100000$ pasos de integración y para el régimen estacionario $BD_{step} = 1000$ pasos de integración.
- En dinámica de Monte Carlo, para el régimen transitorio $MCD_{step} = 10000$ pasos de integración y para el régimen estacionario $MCD_{step} = 1000$ pasos de integración.

III-D. Cálculo de observables

Con el fin de reproducir los resultados publicados en el artículo de Hansen y Verlet se llevaron a cabo simulaciones variando la densidad del sistema en el rango $\rho = [0,8 : 1,1]$, considerando diez valores diferentes de densidad y a una temperatura fija T_{adim} se fueron obteniendo valores medios de presión osmótica, factor de estructura estático y desplazamiento cuadrático medio (y en consecuencia el coeficiente de difusión a tiempos largos). Luego, se cambió la temperatura considerando cuatro valores diferentes $T_{adim} = 0,75; 1,15; 1,35; 2,74$.

Para el cálculo de los valores promedios y errores nos valemos de las propiedades de ergodicidad del sistema el cual nos permite calcular valores de expectación de una única corrida (experimento) suficientemente larga (promedio temporal) y no muchos experimentos con configuraciones

completamente descorrelacionadas entre si (promedio en el ensamble). De esta forma se reduce el tiempo de computo, aunque se obtiene, por cada corrida un único error al final de la misma y en los pasos intermedios se obtienen valores medios que convergen a su valor de equilibrio. Es necesario aclarar que, en todas las simulaciones se debería tener en cuenta la correlación temporal de los observables, este análisis se debería realizar *a posteriori* para mejorar los resultados de las simulaciones, realizando mediciones de observables en tiempos de evolución mayores al tiempo de correlación y asegurarnos así de obtener mediciones independientes, además, este tiempo de correlación nos permitiría conocer bien los errores cometidos.

III-D1. Cálculo de presión osmótica

La presión se calculó teniendo en cuenta únicamente el término correspondiente al teorema del virial de la siguiente manera:

$$P = \frac{1}{dV} \left\langle \sum_{\langle ij \rangle} \vec{f}(\vec{r}_{ij}) \cdot \vec{r}_{ij} \right\rangle \quad (14)$$

$$d = 3; V \equiv L^3 = n_p / \rho$$

Entonces, las gráficas asociadas a cálculos de presión corresponden a la presión osmótica total del sistema. Ahora bien, adimensionalmente y trabajando un poco la ecuación tendremos una expresión para la presión adimensional, la cual fue implementada en las simulaciones

$$P^* = \frac{\rho}{3n_p} \left\langle \sum_{\langle ij \rangle} r_{ij}^* f_{r,ij}^* \right\rangle \quad (15)$$

Notemos que $\langle \dots \rangle$ se refiere al promedio temporal. Además, teniendo en cuenta que para la reducción del tiempo de CPU en las simulaciones se computan los cálculos relacionados a la fuerza de interacción hasta un radio de corte r_{cutoff} sin embargo, el potencial y la fuerza, más allá de este radio no es nula (lo cuál suponemos en los cálculos), entonces, se inducirá un error sistemático respecto a la solución real. En el caso particular de la presión la corrección a la misma se puede calcular de la siguiente manera:

$$\Delta P_{tail} = \frac{16}{3} \pi \rho^2 \epsilon \sigma^3 \left[\frac{2}{3} \left(\frac{\sigma}{r_{cutoff}} \right)^9 - \left(\frac{\sigma}{r_{cutoff}} \right)^3 \right] \quad (16)$$

y de forma adimensional tendremos la siguiente expresión;

$$\Delta P_{tail}^* = \frac{16}{3} \pi \rho^2 \left(\frac{1}{r_{cutoff}^*} \right)^3 \left[\frac{2}{3} \left(\frac{1}{r_{cutoff}^*} \right)^6 - 1 \right] \quad (17)$$

que, en el caso en que, $\rho = 0,8$ y $r_{cutoff}^* = 2,5$ tendremos un tail corrección de $\Delta P_{tail}^* = -0,6844$.

Esta corrección fue tomada en cuenta para plotear los resultados obtenidos de presión en función de la densidad reducida. En la figura 14 se observan los valores.

Por otro lado, para obtener la corrección a la energía potencial podemos usar la siguiente expresión:

$$u_{tail} = \frac{8}{3} \pi \rho \epsilon \sigma^3 \left[\frac{1}{3} \left(\frac{\sigma}{r_{cutoff}} \right)^9 - \left(\frac{\sigma}{r_{cutoff}} \right)^3 \right] \quad (18)$$

y de forma adimensional tendremos la siguiente expresión;

$$u_{tail}^* = \frac{8}{3} \pi \rho \left(\frac{1}{r_{cutoff}^*} \right)^3 \left[\frac{1}{3} \left(\frac{1}{r_{cutoff}^*} \right)^6 - 1 \right] \quad (19)$$

que, en el caso en que, $\rho = 0,8$ y $r_{cutoff}^* = 2,5$ tendremos un tail corrección de $\Delta u_{tail} = -0,4198$. Esta corrección fue tomada en cuenta para plotear las energías mostradas en la figura 13.

Analizando las gráficas mostradas en 14 podemos identificar aquellos valores de densidad donde se produce un salto de discontinuidad en la presión osmótica promedio. Entonces, identificando estos puntos y asociándolos a una temperatura a su temperatura correspondiente se puede realizar el diagrama de fases del sistema.

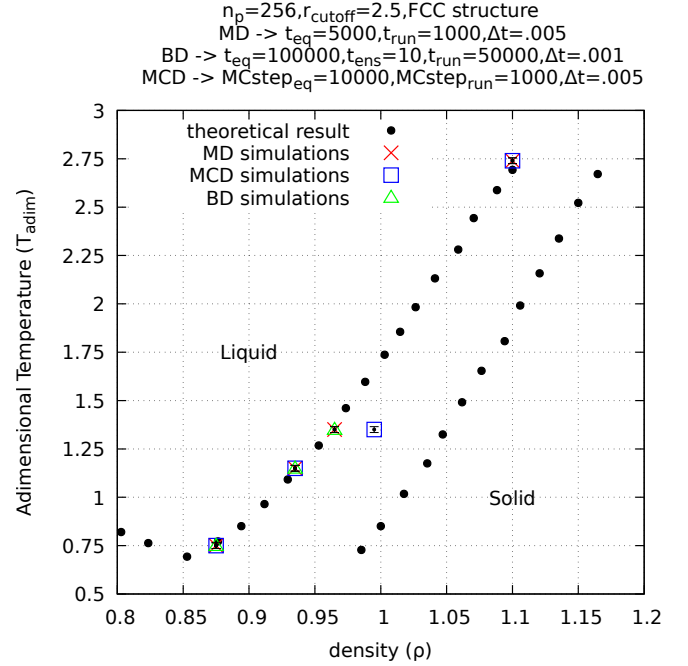


Fig. 8. Temperatura adimensional vs densidad reducida

En la figura 8 se pueden notar que los tres métodos arrojan valores similares a los predichos teóricamente, ya que, los puntos negros corresponden a la ecuación de estado para el sistema de Lennard-Jones donde, tanto a la derecha como a la izquierda de los puntos se obtienen fases bien definidas (líquido a la izquierda y sólido a la derecha) y entre puntos se obtiene una zona de coexistencia de las dos fases (entre límites de fusión y congelación). Por otro lado, podemos observar que como las simulaciones se obtuvieron variando la densidad de forma creciente el sistema se fue comprimiendo modelizando una transformación de líquido a sólido (y no viceversa), por ello, los puntos se encuentran mayormente situados en la línea teórica de congelación (izquierda) y no en la de fusión (derecha).

Finalmente observamos que MCD y BDM obtuvieron algunos puntos del diagrama de fases que no coinciden con DM (el cual arroja resultados en acuerdo con la ecuación de estado teórica).

III-D2. Factor de estructura cristallino

La definición del factor de estructura estático es la siguiente:

$$S(\vec{k}, t) = \frac{1}{N^2} \left| \sum_j \exp(i \vec{k} \cdot \vec{r}_j(t)) \right|^2 \quad (20)$$

$$\vec{k} = \frac{2\pi}{a} (-\hat{e}_x + \hat{e}_y - \hat{e}_z); a = \sqrt[3]{\frac{n_{FCC}}{\rho}}$$

donde $\vec{k}$ es un vector de onda (estrictamente un momento cristalino) perteneciente a la red recíproca, a es el parámetro de red de la red cristalina y $n_{FCC} = 4$ es la cantidad de átomos por celda unidad.

En la figura 15 se puede observar los resultados obtenidos para el factor de estructura estático, el cual es un parámetro de orden cristalino del sistema. Como podemos notar, para el caso del sistema líquido el factor de estructura se anula evidenciando un completo desorden estático del sistema (el tiempo de decaimiento del factor de estructura en DM, para un fluido, está asociado directamente con la dimensión del sistema con una pendiente proporcional $1/n_p$) y para el caso del sistema sólido el factor de estructura se estabiliza en un valor menor a la unidad, lo cual es esperable. Notamos además, que para el rango de densidades estudiado el sistema a la temperatura de referencia adimensional de $T_{adim} = 2,74$ no ocurre una transición de fase, pues el factor de estructura es nulo en todo el rango, evidenciando que el sistema permanece siempre en estado líquido. Notamos también que, para el caso de BD, los resultados no concuerdan con los esperados (específicamente para las temperaturas $T_{adim} = 1,15$ y $T_{adim} = 1,35$ pues el factor de estructura es muy cercano a cero en todo el rango de densidades, lo que evidencia que el sistema se encuentra en estado sólido y cómo sabemos que efectivamente se produce una transición líquido-sólido a estas temperaturas y en estos rangos de densidades, podemos decir que los resultados de las simulaciones no son correctos. Y, para la temperatura $T_{adim} = 0,75$ el factor de estructura según BD induciría a pensar que la transición ocurre para densidades entorno a $\rho = 1,02$ lo que es cierto según la ecuación de estado mostrada en 8, sin embargo, esto debería ocurrir si comenzamos las simulaciones barriendo densidades de forma decreciente lo cual no corresponde con lo realizado. Estas diferencias significativas, podría deberse a un error en el algoritmo que calcula dicho factor de estructura estático.

III-D3. Desplazamiento cuadrático medio y coeficiente de difusión

La definición de desplazamiento cuadrático medio (MSD) y su relación con el coeficiente de difusión es la siguiente:

$$\langle |\vec{r}(t) - \vec{r}(0)|^2 \rangle = \frac{1}{n_p} \sum_{i=1}^{n_p} [\Delta \vec{r}_i(t)]^2 \cong 6Dt \quad (21)$$

Estrictamente hablando, la relación anterior en que se cumple que el MSD evoluciona lineal con el tiempo de correlación, es válida únicamente en el límite de tiempos largos. Donde D es el coeficiente de difusión, es importante mencionar que para el cálculo de MSD es necesario computar las autocorrelaciones de pares utilizando las posiciones sin corregir según PBC, para ello deberemos guardar a todo paso de integración los vectores posición de las partículas sin aplicación de PBC. Además, teniendo en cuenta que las únicas dinámicas realistas del presente trabajo son la dinámica molecular y la dinámica Browniana, solo con estas se procedió a calcular el coeficiente de difusión a tiempos largos, para ello se consideraron $\tau_{corr}^{max} = 10000$ pasos máximos de correlación para el cómputo del MSD.

III-E. Coeficiente de difusión a tiempos largos

Los parámetros elegidos para este caso fueron $n_p = 256$ partículas, radio de corte de $r_{cutoff} = 2,5$, temperaturas simuladas con valores de $T_{adim} = 0,75, 1,15, 1,35, 2,74$. Además, se varió la densidad entre los valores $\rho_{min} = 0,8$

y $\rho_{max} = 1,1$ (realizando 10 simulaciones con diferentes densidades dentro de dicho rango) y, según la dinámica elegida, tendremos los siguientes parámetros de simulación:

- En dinámica molecular, $MD_{step} = 5000$ pasos de equilibración (régimen transitorio) y $MD_{step} = 50000$ pasos de corrida (régimen estacionario). Un paso temporal de $\Delta t = 0,005$.
- En dinámica Browniana, $BD_{step} = 100000$ pasos de equilibración (régimen transitorio) y $BD_{step} = 50000$ pasos de corrida (régimen estacionario). Un paso temporal de $\Delta t = 0,001$.

y $\tau_{corr}^{max} = 10000$ pasos máximos de correlación para el cómputo del MSD para ambas dinámicas. (notemos que si bien los pasos máximos de correlación son idénticos en ambas simulaciones el tiempo real máximo de correlación no es igual pues $t_{corr,real}^{max} = \Delta t \cdot \tau_{corr}^{max}$).

Para dinámica Browniana en la figura 16 se pueden apreciar los resultados obtenidos, donde se puede notar la dependencia del coeficiente de difusión adimensionalizado D/D_0 con el tiempo de correlación adimensionalizado t/t_0 (donde $t_0 = \sigma^2/D_0$ y D_0 corresponde al coeficiente de difusión a tiempos cortos) está en acuerdo con la teoría, es decir, observando el gráfico de MSD vs tiempo de correlación se identifica primero un proceso difusivo a tiempos cortos (tiempos comparados con τ_a), luego un proceso sub-difusivo (evolución con potencias temporales menores a la unidad, es decir, menor pendiente que la relación lineal) y un proceso difusivo para tiempos largos (evolución lineal). Mientras que para dinámica molecular en la figura 17 se pueden apreciar los resultados obtenidos, donde se puede notar la dependencia del coeficiente de difusión D con el tiempo de correlación t que está en acuerdo con la teoría, es decir, en el gráfico de MSD vs tiempo de correlación se identifica un proceso superdifusivo a tiempos cortos (evolución con potencias temporales mayores a la unidad) y un proceso difusivo para tiempos largos (evolución lineal).

Para la determinación de los coeficientes de difusión a tiempos largos de correlación se utilizan las gráficas de coeficiente de difusión vs tiempo de correlación, donde esperamos que el valor buscado corresponda a la constante en la cual se aplanan la curva para tiempo largos. Lamentablemente, sólo fue posible hacer este análisis con los resultados obtenidos de DM pues se cuenta con un gran rango de tiempos de correlación para la cual el coeficiente de difusión se mantiene constante (gran zona de proceso difusivo), sin embargo, en BD no fue posible obtener una zona muy grande debido a una mala elección de tiempo máximo de correlación (debería haberse elegido un valor mucho mayor a 50000 pasos de integración en régimen estacionario para considerar tiempos de correlación máximos mayores a los 10000 pasos empleados, sin embargo, teniendo en cuenta que el régimen transitorio considerado fue de 100000, el coste computacional aumentaría demasiado).

Los resultados obtenidos se muestran en la figura 9, donde se pueden observar saltos en el coeficiente de difusión en los rangos de densidades en que se produce la transición líquido-sólido (según mostramos en 8).

III-F. Análisis de performance

En las simulaciones de MCD, es importante notar que el tiempo de cpu se reduce notablemente si procedemos a calcular la diferencia de energía potencial de alguna forma que nos permita solamente tener en cuenta la partícula que

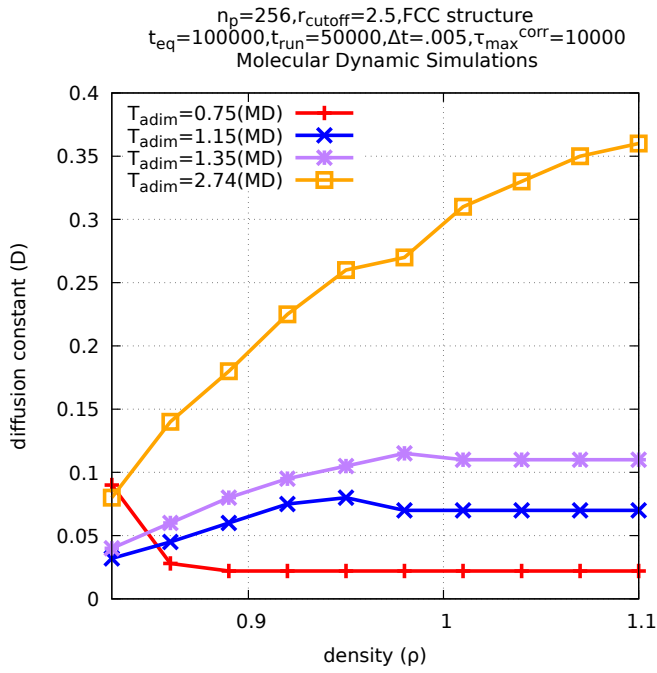


Fig. 9. Coeficiente de difusión y MSD vs densidad

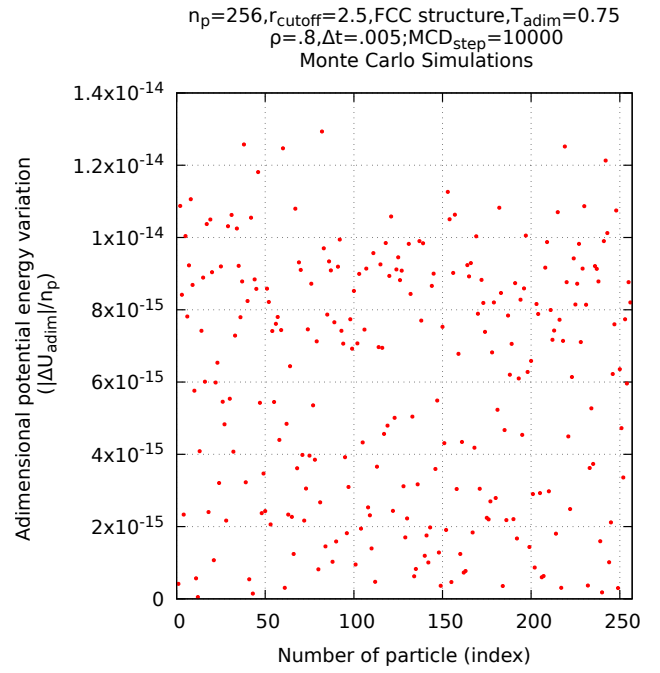


Fig. 10. ΔU vs índice de partícula

se está desplazando, en lugar, de considerar las posiciones de todas las partículas cada vez y luego emplear el algoritmo de metrópolis en cada paso de MCD. El código desarrollado para emplear esta mejora se muestra en 6 donde notamos que el loop original de doble ciclo for (o do) anidado se reduce por un único loop for (o do), reduciendo el tiempo en un orden n_p . Además, a fines de comprobar que ambos métodos eran equivalentes, procedimos a graficar la restas de ambas expresiones que nos permiten calcular la diferencia de energía, por un lado la que requiere el cálculo explícito de toda la energía potencial en los dos pasos (aceptando y descartando el desplazamiento) y por otro lado la expresión que requiere únicamente los términos del potencial que involucran a la partícula desplazada. En la figura 10 se pueden ver las diferencias entre estos dos cálculos y vemos que los valores se encuentran dentro del orden de precisión de la máquina 10^{-14} para doble precisión, resultado que nos deja tranquilos en saber que estamos utilizando métodos equivalentes.

Entonces, para notar explícitamente la reducción drástica de tiempo de cpu, se realizó un escalamiento del problema realizando corridas del algoritmo para distintos tamaños (número de partículas totales) para una dada temperatura. Recordando que para una estructura inicial FCC el número de partículas debe definirse de la forma $n_p = 4 \cdot i^3; i \in \mathbb{N}$, en nuestro caso se emplearon partículas con $2 \leq i \leq 5$. Para definir el radio de corte de interacciones se podría tomar como criterio que el mismo sea igual a $r_{cutoff} = 0,3 \cdot L < 0,5 \cdot L$ donde $L = \sqrt[3]{\frac{n_p}{\rho}}$ es el lado del cubo de estudio el cual se repite según las PBC, sin embargo, a los fines prácticos de medir el coste computacional se mantuvo un radio de corte fijo de $r_{cutoff} = 2,5$ para todos los casos.

Listing 6: Subroutine: Cambio de energía

```
1 ! compute total lennard jones potential (TRUNCADO Y
  DESPLAZADO)
2 function u_lj_reduced(n_p, x_vector, y_vector, z_vector
  , r_cutoff, density, index)
3 integer(sp), intent(in) :: n_p, index
4 real(dp), intent(in) :: x_vector(n_p), y_vector(n_p), z_vector(n_p)
```

```
    n_p, z_vector(n_p)
    real(dp), intent(in) :: r_cutoff, density
    real(dp) :: u_lj_reduced, u_indiv,
    rij_pow02
    integer(sp) :: j ! indices para
    etiquetar par de partículas
    u_lj_reduced=0._dp
    do j=1, n_p
        if (j/=index) then ! i=index
            ! calculamos distancia relativa corregida segun
            PBC
            rij_pow02=rel_pos_correction(x_vector(index),
            y_vector(index), z_vector(index), &
            x_vector(j), y_vector(j), z_vector(j), n_p, density)
            if (rij_pow02==0._dp) then; print *, 'rij_pow02=0';
            stop; end if
            if (rij_pow02 <= r_cutoff*r_cutoff) then
                u_indiv=u_lj_individual(rij_pow02)!-
                u_lj_individual(r_cutoff*r_cutoff)
                u_lj_reduced=u_lj_reduced+u_indiv
            end if; end if; end do
    end function u_lj_reduced

21 function delta_u_lj(n_p, x_vector, y_vector, z_vector,
    r_cutoff, density, x_value, y_value, z_value, index)
    ! numero de partículas e índice de la partícula
    desplazada
    integer(sp), intent(in) :: n_p, index
    real(dp), intent(inout) :: x_vector(n_p),
    y_vector(n_p), z_vector(n_p)
    ! valores posicion originales
    real(dp), intent(in) :: x_value, y_value,
    z_value
    real(dp), intent(in) :: r_cutoff, density
    real(dp) :: delta_u_lj
    real(dp) :: x_value_new,
    y_value_new, z_value_new
    ! guardamos las posiciones desplazadas
    x_value_new=x_vector(index)&
    y_value_new=y_vector(index); z_value_new=z_vector(
    index)
    delta_u_lj=0._dp
    ! computamos energia con posiciones sin desplazar (
    valores originales)
    x_vector(index)=x_value; y_vector(index)=y_value;
    z_vector(index)=z_value
    delta_u_lj=delta_u_lj-&
    u_lj_reduced(n_p, x_vector, y_vector, z_vector,
    r_cutoff, density, index)
    ! computamos energia con posiciones desplazadas
    x_vector(index)=x_value_new&
```

```

40 y_vector(index)=y_value_new; z_vector(index)=
    z_value_new
41 delta_u_lj=delta_u_lj+&
42 u_lj_reduced(n_p, x_vector, y_vector, z_vector,
    r_cutoff, density, index)
43 end function delta_u_lj

```

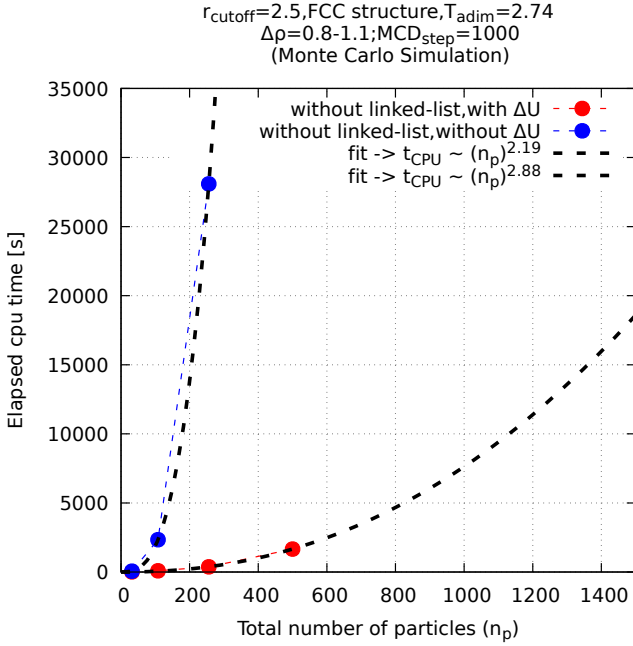


Fig. 11. Número total de partículas vs tiempo de CPU

En la gráfica 11 se observan los resultados obtenidos, en donde se llevó a cabo un ajuste de datos a través de una ecuación polinómica mediante dos parámetros de ajuste, y en donde podemos observar que el coste computacional escala aproximadamente de forma cuadrática con el número de partículas para el mejor método (utilizando subrutina delta_u_lj sin linked-list) y escala aproximadamente de forma cúbica con el número de partículas para el peor método (sin utilizar subrutina delta_u_lj y sin linked-list).

Por otro lado, para lograr un escalamiento lineal del coste computacional se emplean mejoras algorítmicas como las llamadas **linked list**, estas listas son las comúnmente utilizadas en cualquier simulación de DM y BD. Es posible también emplear estas técnicas en MCD sin embargo, la ganancia computacional no se mejora notablemente a partir de un algoritmo que implemente una subrutina específica en el cálculo de la variación de energía (ver curvas mostradas en 11).

En las simulaciones de DM y BD se implementaron estas listas anidadas para computar tanto las fuerzas, como energías y presiones en las simulaciones. Para observar las ganancias computacionales se midieron los tiempos de cpu para simulaciones de DM que utilizan el método de optimización linked-list contra aquellas simulaciones que no la utilizan. Cabe mencionar que la ganancia en tiempo de cpu utilizando linked-list se nota apreciablemente a medida que aumentamos el tamaño del sistema pues, el parámetro m (que nos permite dividir en subceldas más pequeñas la celda de simulación original de lado L) puede aumentar notablemente según la restricción $m \leq \frac{L}{\text{anint}(\frac{L}{r_c})}$. En la figura 12 se observan los resultados obtenidos donde vemos que el método utilizando linked-list escala con potencias del número de partículas

menores a dos y el método sin utilizar linked-list escala con potencias de aproximadamente polinómicas.

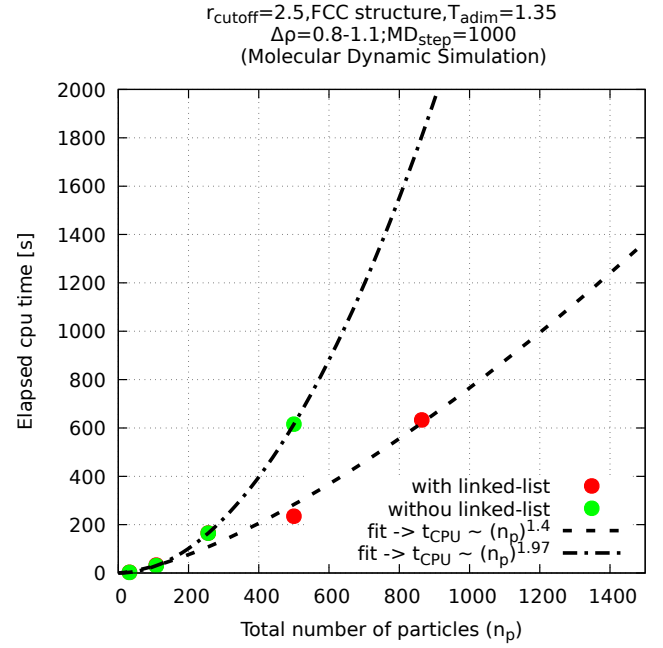


Fig. 12. Número total de partículas vs tiempo de CPU

Por otro lado, observamos que existen algunas ligeras diferencias respecto a los resultados teóricos que nos dicen que se debería tener una mejora de escalamiento n_p^2 a escalamiento lineal n_p^1 . Las diferencias, podrían deberse a que no se realizó un análisis exhaustivo de performance, realizando simulaciones con muchos tamaños de partículas y realizando estadística. Además, las diferencias podrían ocurrir por una mala implementación de las listas anidadas, o una no tan eficiente, por ejemplo, en el presente trabajo no se utilizaron punteros para diseñar estos algoritmos, lo cual sabemos impacta sustancialmente en la performance del mismo, trabajando de forma más eficiente con los registros.

Los algoritmos desarrollados para las tres dinámicas que utilizan linked-list fueron

- f_lj_total_linkedlist
- u_lj_total_linkedlist
- osmotic_pressure_linkedlist
- maps
- icell_function
- links
- evolution_bd_linked_list
- velocity_verlet_linked_list
- evolution_monte_carlo_linkedlist

que se encuentran dentro de los módulos **module_md_lennard_jones.f90**, **module_bd_lennard_jones.f90**, **module_mc_lennard_jones.f90**.

III-G. Conclusiones generales

En términos generales, se ha reproducido los valores del artículo, específicamente, los asociados a la ecuación de estado del sistema de Lennard-Jones, en el cual, se pudo identificar, en el rango admisible, la transición de fase líquido-sólido del sistema. Sin embargo, se han obtenido ciertas diferencias entre los resultados de cada método, por ello, es necesario aclarar que si bien los valores de equilibrio (estacionarios) de este sistema en partícula no depende de

la dinámica, pues todas las dinámicas nos han permitido reproducir los valores de energía, presión, factor de estructura y coeficiente de difusión a tiempos largos, las diferencias podrían deberse a una incorrecta aplicación de los métodos pues, detectamos que en BD deberíamos requerir aún más pasos de equilibración y evolución del sistema para reproducir los datos asociados al coeficiente de difusión a tiempos largos, mientras que, en conjunto con MCD, se obtuvieron algunas discrepancias en algunos puntos del diagrama de fases de la figura 8. Según algunos estudios las diferencias entre valores teóricos y simulaciones se pueden deber principalmente a los efectos de tamaño (pocos números de partículas) y truncamiento del potencial (se considera un radio de corte fijo para todas las simulaciones y tamaños). Por otra parte la figura 1 del artículo de Hansen y Verlet muestra algunas diferencias significativas respecto a los valores obtenidos de simulaciones, pues los autores mencionan un método más sofisticado para la obtención de los valores de presión (cell-modelz "homogenized fluid model") en función de la densidad reducida. Además, la figura 6 del artículo, que muestra la variación del factor de estructura en función de la densidad, no se pudo reproducir teniendo en cuenta que este observable corresponde al factor de estructura para líquidos y gases, que requiere mayor estadística para su obtención. En el presente trabajo se obtuvo el factor de estructura del sólido el cual notamos que difiere notablemente del mostrado en el artículo. Sin embargo, cabe resaltar que los autores Hansen y Verlet encontraron que para sistema de tipo Lennard-Jones la línea de transición líquido-sólido se presenta cuando el factor de estructura corta el valor de aproximadamente 2,85, lo cual es una técnica útil y rápida de visualizar la transición.

Finalmente, podemos decir que a bajas densidades se espera que MCD funcione bien pues los cambios configuracionales se pueden realizar sin problemas (aunque estas evoluciones no son físicamente realistas) y el sistema equilibra a los valores esperados. En cambio a densidades altas los cambios configuracionales no pueden realizarse con facilidad y los resultados de equilibrio, en general, no son correctos.

Los resultados obtenidos con DM estuvieron muy en acuerdo con los mencionados en el artículo de referencia y podemos decir que para este caso particular es el más conveniente por requerir menor coste computacional para la obtención de los valores a reproducir. Y que, el método de BD requiere mucho esfuerzo computacional para una reproducibilidad de los resultados (por ejemplo, en el presente trabajo se utilizaron 150000 pasos de integración y aún así el coeficiente de difusión a tiempos largos no se estabilizó). El método de MCD se encuentra en un nivel intermedio, sin embargo, hay que recordar siempre que la dinámica de este método no es realista por ello, propiedades como el coeficiente de difusión a tiempos largos, que requiere una reproducción fiel y físicamente correcta de la dinámica no se pueden obtener mediante MCD y sólo pueden calcularse con BD o DM.

Finalmente, debemos mencionar que es muy importante analizar la autocorrelación temporal debido a que el sistema no solo está regido por correlaciones espaciales sino también por correlaciones temporales que, si estas son significativas, los errores estadísticos de un dado observable (que está autocorrelacionado temporalmente) se verán incrementados en un factor $\sqrt{2\tau_{corr}}$, además, el cálculo de autocorrelación temporal nos brinda información respecto a la existencia de observables que están relacionados dinámicamente a tiempos largos.

REFERENCIAS

- [1] E. Flenner and G. Szamel, Relaxation in a Glassy Binary Mixture: Comparison of the Mode-Coupling Theory to a Brownian Dynamics Simulation, *Phys. Rev. E* 72, 031508 (2005).
- [2] Stokes' Law, in Wikipedia (2022).
- [3] Teorema de fluctuación-disipación, in Wikipedia, la enciclopedia libre (2022).
- [4] G. Nägele, Brownian Dynamics simulations, Institut für Festkörperforschung Forschungszentrum Jülich GmbH
- [5] E. Braun, Un movimiento en zigzag (2011).
- [6] Lennard-Jones Potential, in Wikipedia (2022).
- [7] J. Karl Johnson, John A. Zollweg, Keith E. Gubbins (1993): The Lennard-Jones equation of state revisited, *Molecular Physics: An International Journal at the Interface Between Chemistry and Physics*, 78:3, 591-618.
- [8] Hansen, J.P. and Verlet, L., 1969. Phase transitions of the Lennard-Jones system. *physical Review*, 184(1), p.151.
- 9 Hoover, W. G., Ree, F. H. (1968). Melting transition and communal entropy for hard spheres. *The Journal of Chemical Physics*, 49(8), 3609-3617.
- 10 Hoover, W. G., Ree, F. H. (1967). Use of computer experiments to locate the melting transition and calculate the entropy in the solid phase. *The Journal of Chemical Physics*, 47(12), 4873-4878.
- 11 Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6), 1087-1092.
- 12 Shankar, K. N. (2019). A Comparison Between Molecular Dynamics and Monte Carlo Simulations of a Lennard-Jones Fluid CBE 641: Nanoscale Transport Project Report.
- 13 Madurga, S., Pujadas, F. M. (2016). European Master in Theoretical Chemistry and Computational Modelling.
- 14 Frenkel, D., Smit, B. (2001). *Understanding molecular simulation: from algorithms to applications* (Vol. 1). Elsevier.
- 15 Notas de clase, presentaciones y bibliografía del curso de Física Computacional 2022 - FaMAF-UNC

IV. CÓDIGOS

Repositorio de GitHub

- <https://github.com/mendzmartin/fiscomp2022.git>

Repositorio GitHub del trabajo final

- https://github.com/mendzmartin/fiscomp2022/tree/main/final_project

Enlaces a programas principales, Makefile y módulos

- **Programas principales**
- https://github.com/mendzmartin/fiscomp2022/blob/main/final_project/code/molecular_dynamic_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/final_project/code/brownian_dynamic_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/final_project/code/monte_carlo_dynamic_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/final_project/code/Makefile
- **Video de dinámica molecular**
- <https://github.com/mendzmartin/fiscomp2022/blob/main/lab05/prob01/plots/movie.mp4>
- **Video de dinámica browniana**
- <https://github.com/mendzmartin/fiscomp2022/blob/main/lab06/prob01/plots/movie.mp4>
- **Módulos de funciones y subrutinas**
- https://github.com/mendzmartin/fiscomp2022/blob/main/modules/module_md_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/modules/module_bd_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/modules/module_mc_lennard_jones.f90
- https://github.com/mendzmartin/fiscomp2022/blob/main/modules/module_precision.f90

- https://github.com/mendzmartin/fiscomp2022/blob/main/modules/module_mt19937.f90

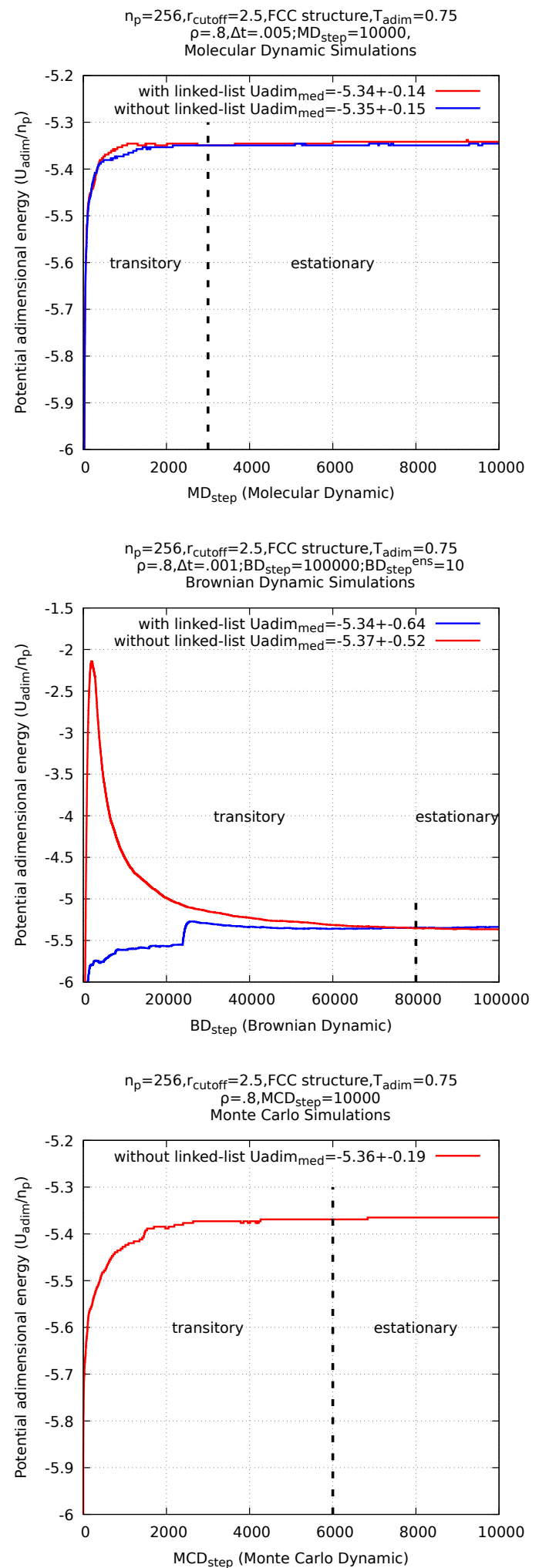


Fig. 13. Energía potencial vs pasos de evolución

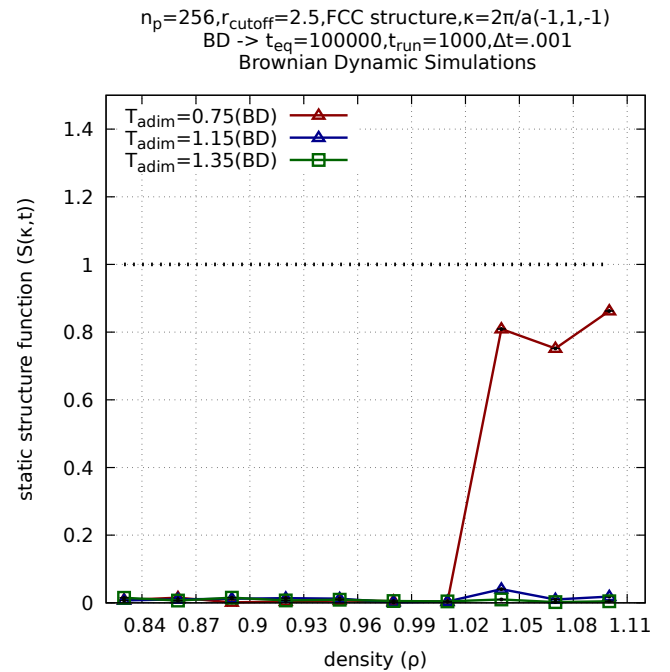
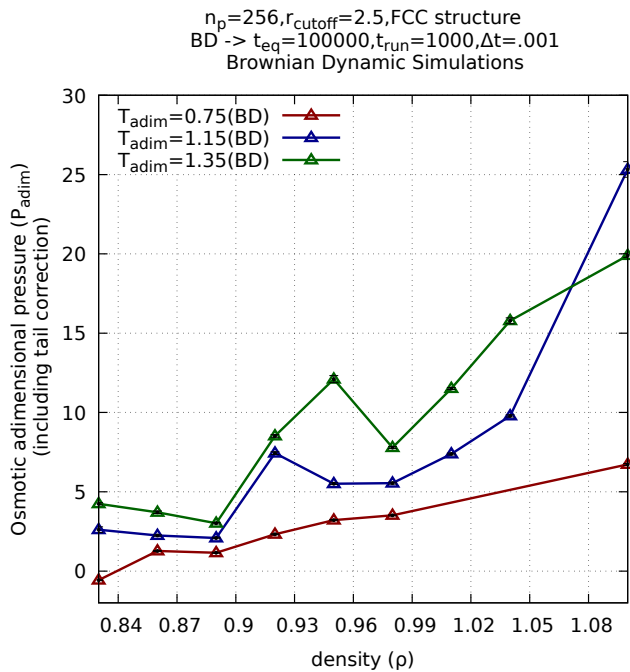
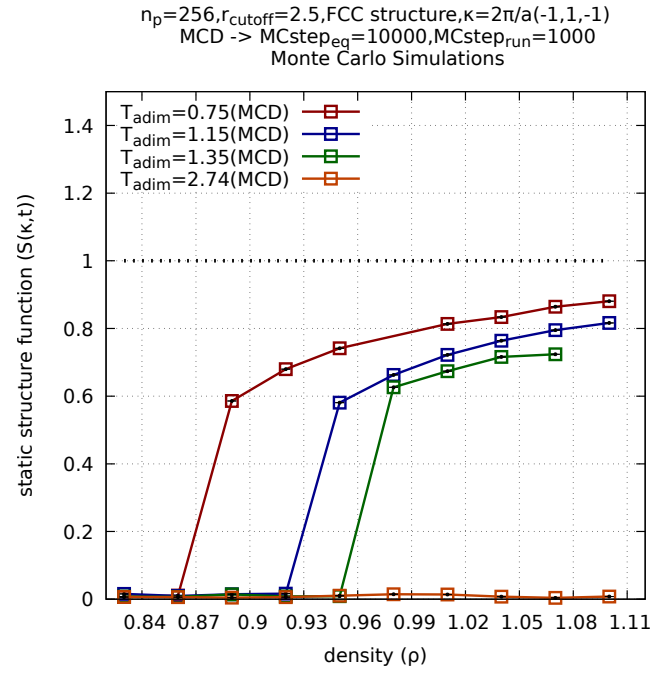
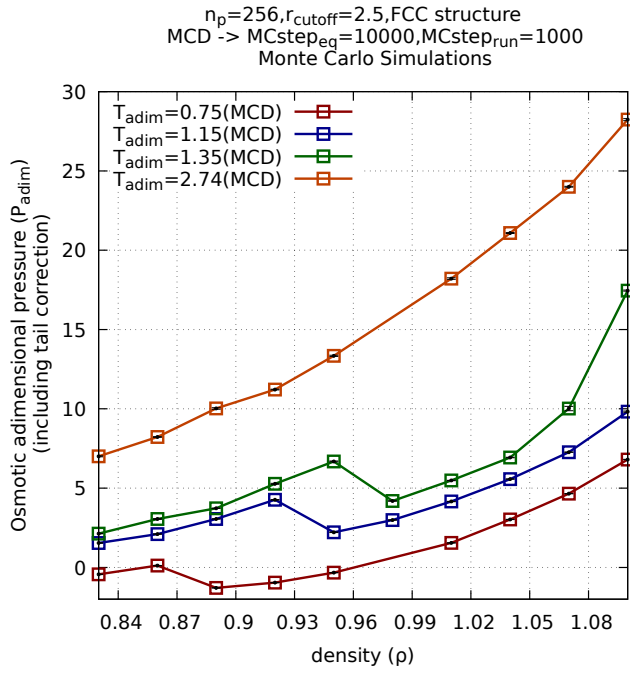
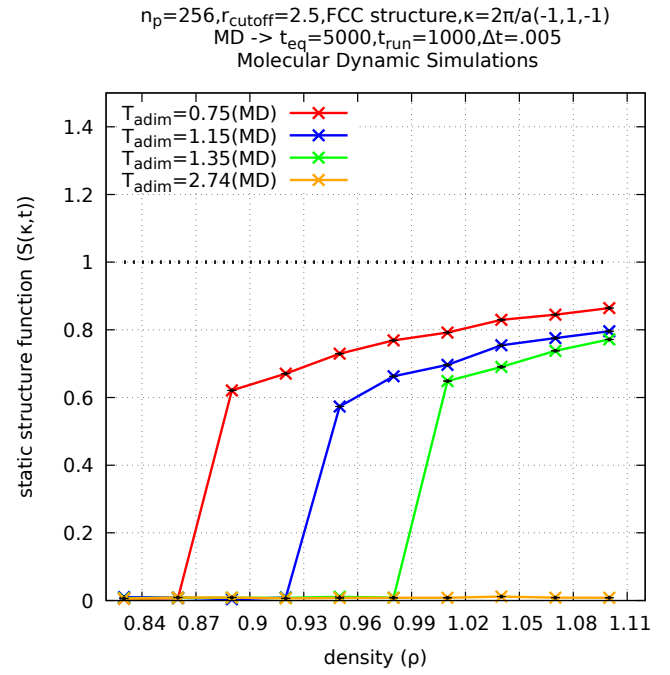
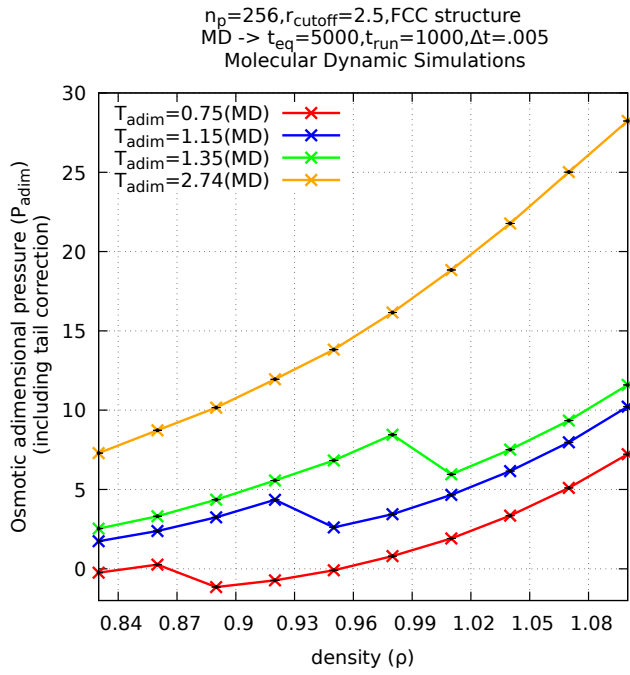


Fig. 14. Presión osmótica vs densidad reducida

Fig. 15. función de estructura estática vs densidad adimensional

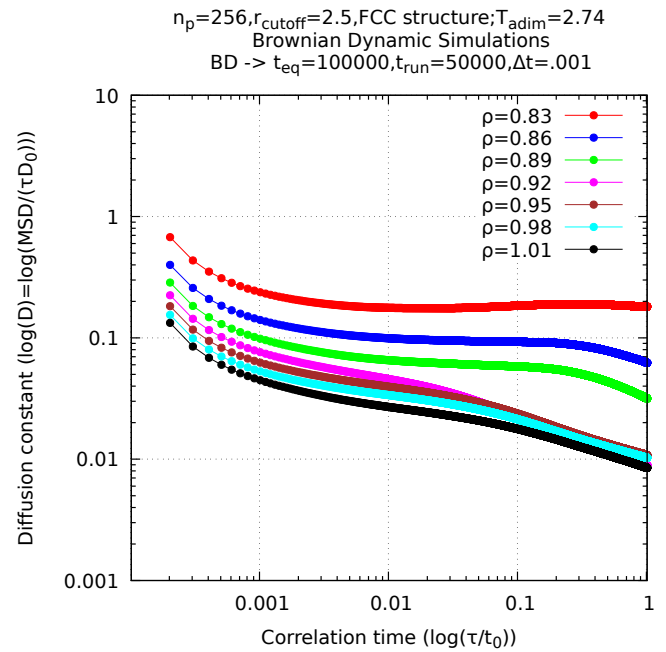
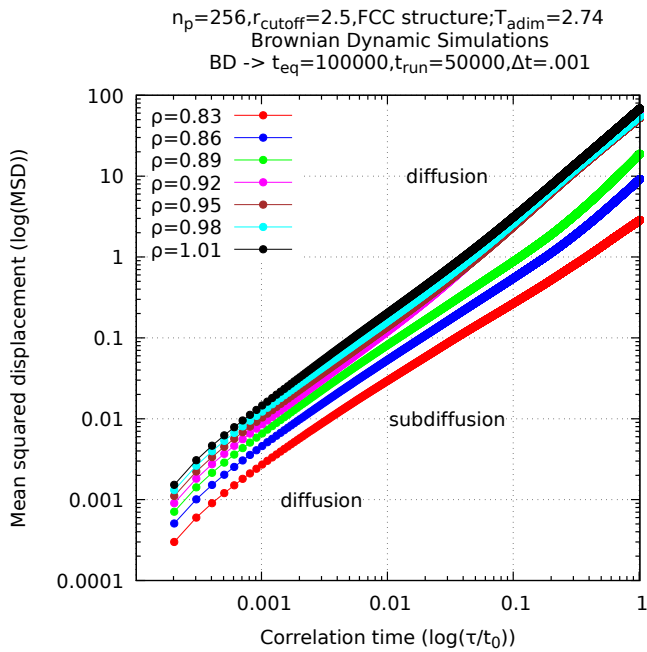
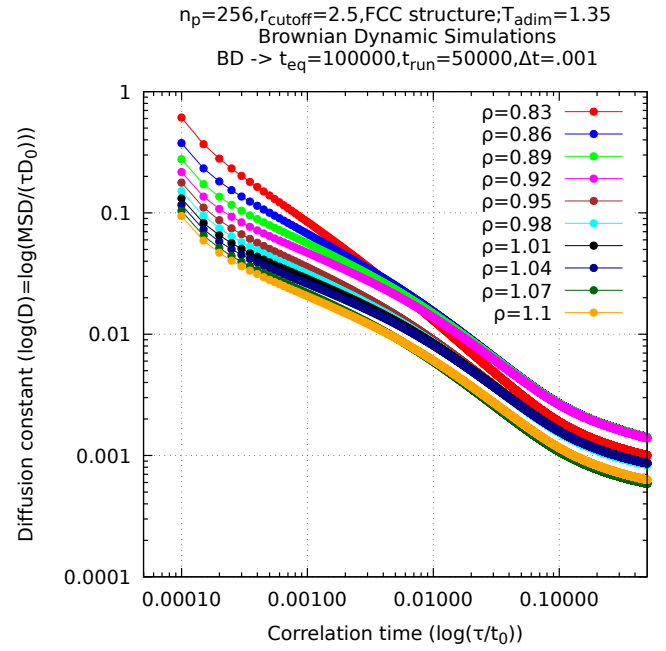
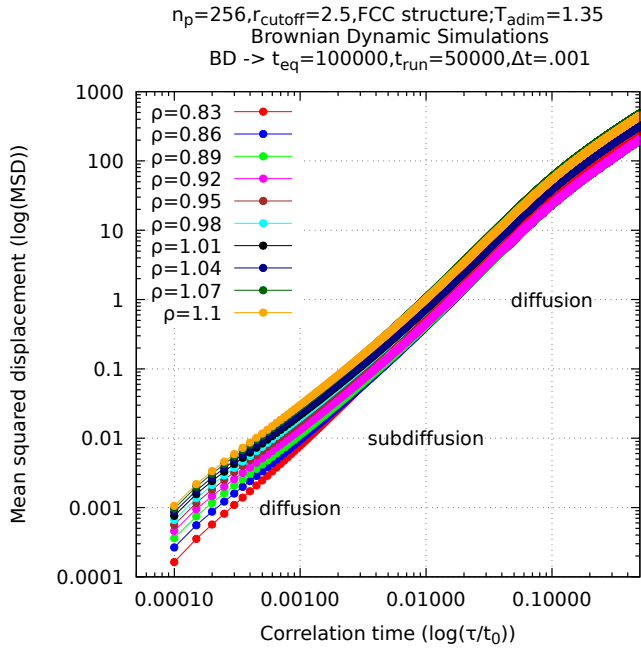
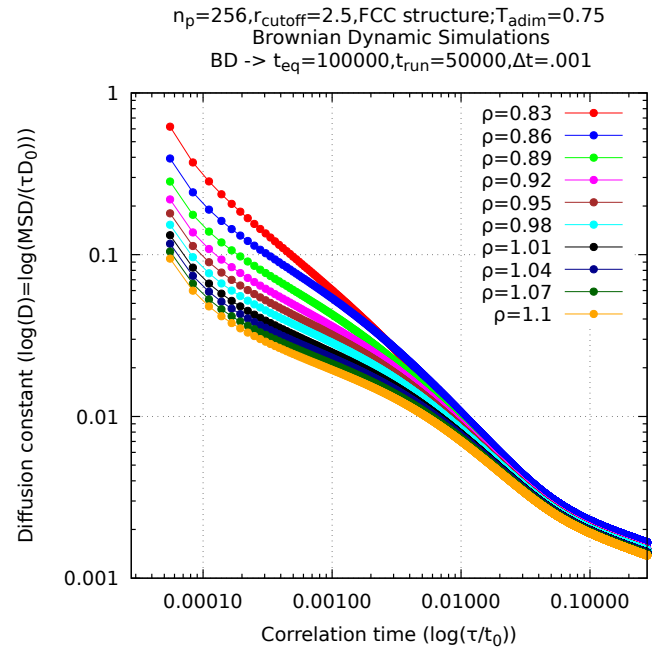
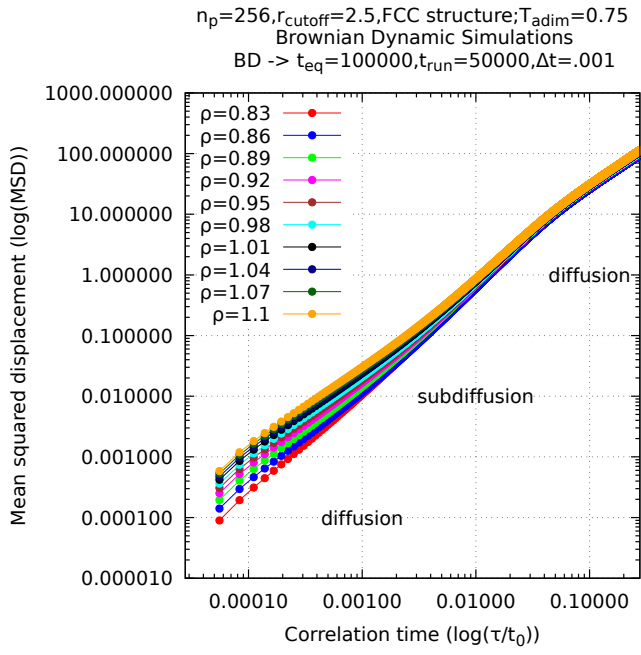


Fig. 16. Coeficiente de difusión y MSD adimensionales vs tiempo de correlación adimensional

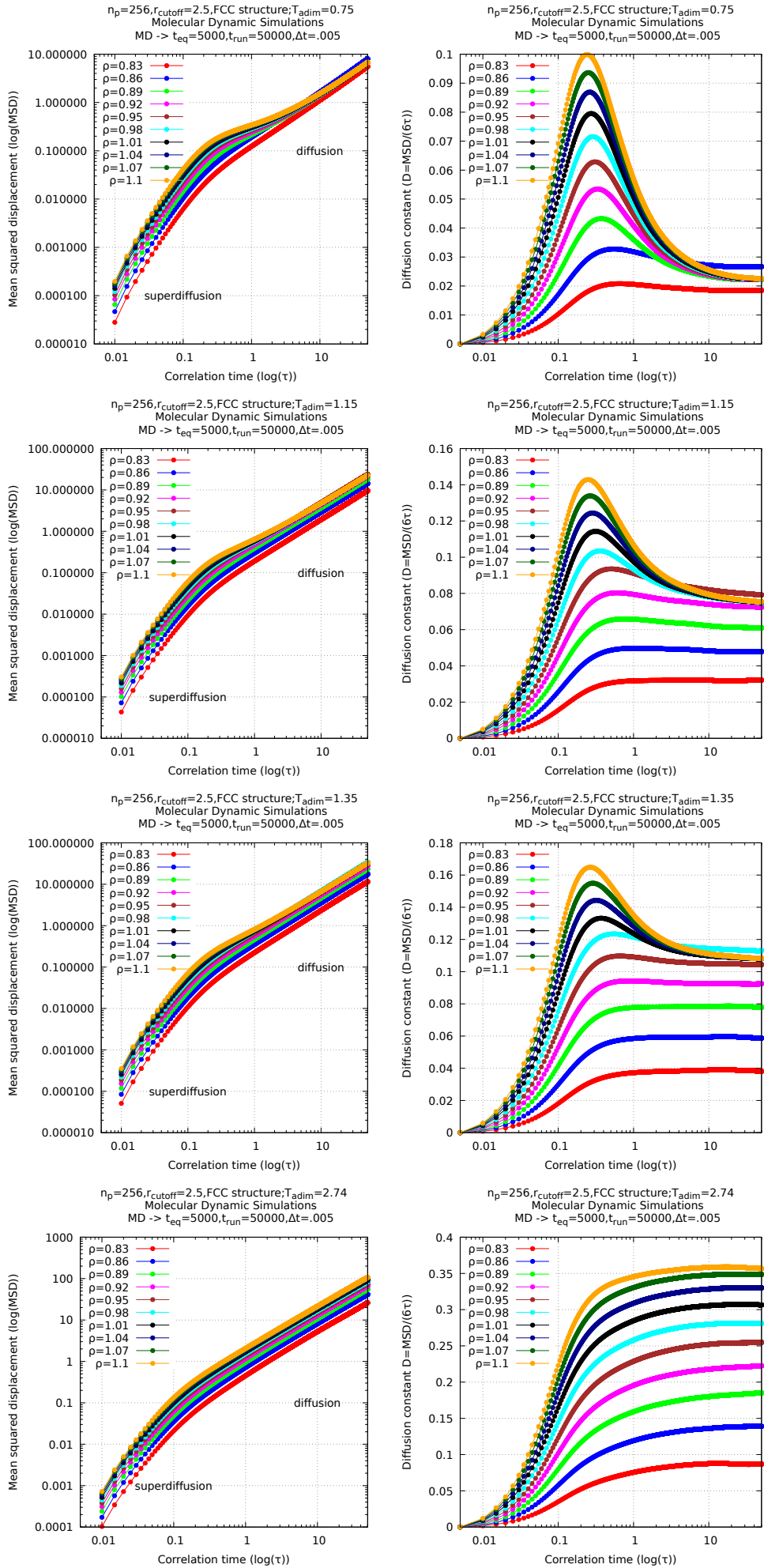


Fig. 17. Coeficiente de difusión y MSD adimensionales vs tiempo de correlación adimensional

Fig. 18. Animación de dinámica Browniana