

# SIMULATIONS WITH CLASSICAL PARTICLES

Jakub W. Narojczyk and Krzysztof W. Wojciechowski

President St. Wojciechowski PWSZ in Kalisz,  
Nowy Swiat 4, 62-800 Kalisz, Poland

Institute of Molecular Physics  
Polish Academy of Sciences  
M. Smoluchowskiego 17, 60-179 Poznan, Poland

Computational Nanotechnology  
2011 Gdansk, Poland  
Lecture 3

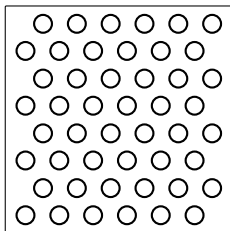
- 1 Modelling the system
  - Concept of a periodic box.
  - Nearest image convention
  - Interaction cut-off
  - Verlet's list
- 2 Studying the system
  - Radial distribution function
  - Correlations
  - Auto-correlation function

# The initial layout

Initialisation of the system under study

## Example implementation

```
for( j = 0; j < Ny; j++ ){  
  for( i = 0; i < Nx; i++ ){  
    x[ i+j*Nx ] = x0 + i * dx;  
    y[ i+j*Nx ] = y0 + j * dy;  
    // shift every even row to the right  
    if ( j & 1 != 0 )  
      x[ i+j*Nx ] += shift;  
  }  
}
```

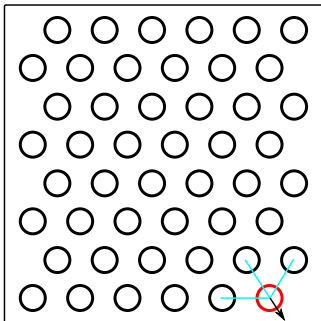


Initial particles' positions. The hexagonal lattice.

for hexagonal lattice

$$dx = 1, dy = \sqrt{3}/2$$

# Boundary effect



We would like to study properties of a large system by examining a smaller part that exhibits those properties

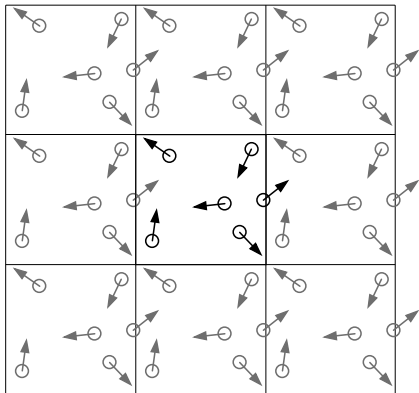
## QUESTION

Is our system homogeneous?

Is every particle in the same conditions?

# Periodic boundary conditions

To minimise the effect of the system's boundary we can use the *Born–Karman* boundary concept.

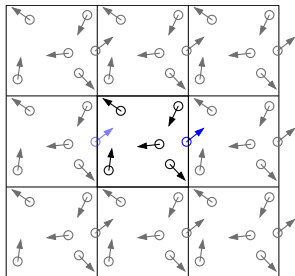


In computer simulations these are called:

**PERIODIC BOUNDARY  
CONDITIONS**

# Implementation of periodic boundary conditions

Instead of making 9 copies of the system (26 in 3D case) we can *wrap the space of our system* in all directions.



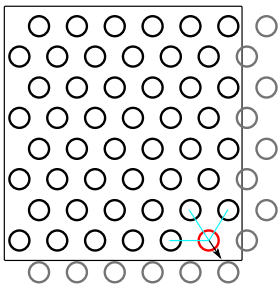
## Example implementation

```
// horizontal conditions
if (x[k] >= box_x/2.0) x[k] = x-box_x;
if (x[k] < -box_x/2.0) x[k] = x+box_x;
// vertical conditions
if (y[k] >= box_y/2.0) y[k] = y-box_y;
if (y[k] < -box_y/2.0) y[k] = y+box_y;
```

The above example assumes that the coordinate system's origin is placed in the centre of the simulated system.

## QUESTION

How to consider interactions for molecules on the edge of the system?



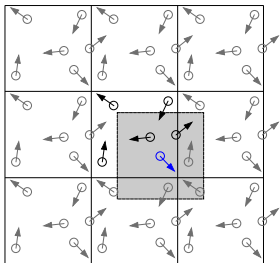
Since we do have the surrounding images of particles, let's allow interactions with the base particles.

## Recepie

Instead of calculating the interactions for every particle only inside the box, let's calculate the interaction with the true particle *or* its periodic image, *which ever is closer* to our particle.

# Interactions with periodic images

This method is referred to as the **MINIMUM IMAGE CONVENTION**



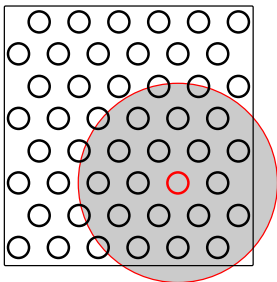
## Example implementation

```
int fx, fy;  
rx = x[m] - x[k];  
ry = y[m] - y[k];  
if ( rx ) fx = rx / fabs(rx);  
if ( ry ) fy = ry / fabs(ry);  
  
if ( fabs( rx ) > box_x/2 )  
    rx = x[m] - x[k] - fx*box_x;  
if ( fabs( ry ) > box_y/2 )  
    ry = y[m] - y[k] - fy*box_y;
```

# How to speed things up?

## QUESTION

Is it really necessary to calculate interactions with *all* the particles in the system?



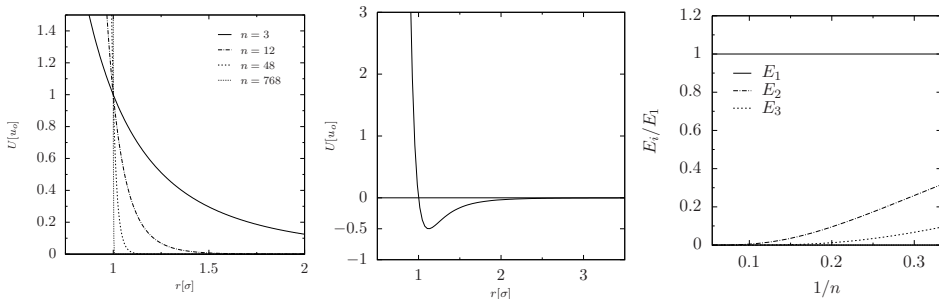
Calculation of forces for all atoms requires the computation of

$$N(N - 1)/2$$

unique distances. Would it be possible to consider only interactions only within the certain area around the particle instead of the whole system?

An answer for this question will reacquire the examination of the interaction potential.

# Truncation of the potential



It is possible to *truncate the interaction* at some distance  $r = r_c$  when the influence of the truncated part becomes insignificant.

However, if the intermolecular potential is not negligible at  $r_c$  (the radius of the cut-off is too small), truncating the interactions will result in a *systematic error*.

# Which atoms are close enough?

Truncation of the potential means that at the distance  $r > r_c$ , the interaction is considered to be zero

## QUESTION

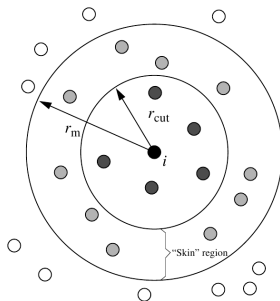
Calculating the distance for pairs of particles for which the interaction will be zero is clearly a *waste of time*. But how to keep track which particles are eligible for calculations?

Answer to this problem is maintaining a list of neighbouring atoms, so called **VERLET'S LIST**

It is a list of all particles  $j$  that are closer to a chosen particle  $i$  than a given distance  $r_m$ , where  $r_m > r_c$ .

The list update interval ( $N_m$ ) and the radius are chosen such that:

$$r_m - r_c > N_m \bar{v} \Delta t$$



# Studying the system's properties

In molecular dynamics we have a number of tools to help visualise the complex data obtained for studied systems and their evolution in time. One such tool is the **RADIAL DISTRIBUTION FUNCTION** (RDF).

The RDF  $g(x)$  measures how atoms organise themselves around one another – it measures the *local structure*. It is proportional to the probability of finding an atom in the distance  $r + \Delta r$  from another atom

## DEFINITION

The radial distribution function is defined by:

$$\rho g(\mathbf{r}) = \frac{1}{N} \left\langle \sum_i^N \sum_{j \neq i}^N \delta[\mathbf{r} - \mathbf{r}_{ij}] \right\rangle$$

where  $\rho = N/V$

# Radial Distribution Function

RDF is of fundamental importance in thermodynamics because macroscopic thermodynamics quantities can be calculated using  $g(r)$ . For example:

- the virial equation for pressure

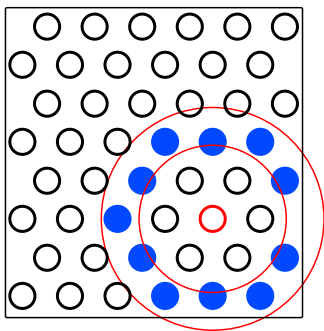
$$p = \rho kT - \frac{2\pi}{3} \rho^2 \int dr \cdot r^3 u'(r) g(r, \rho, T)$$

- the energy equation

$$\frac{E}{NkT} = \frac{3}{2} + \frac{\rho}{2kT} \int dr 4\pi r^2 u(r) g(r, \rho, T)$$

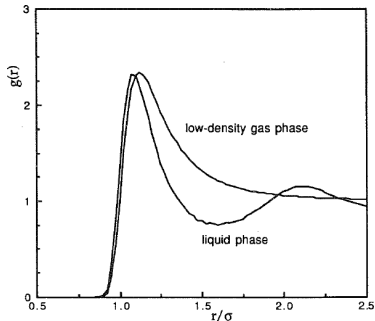
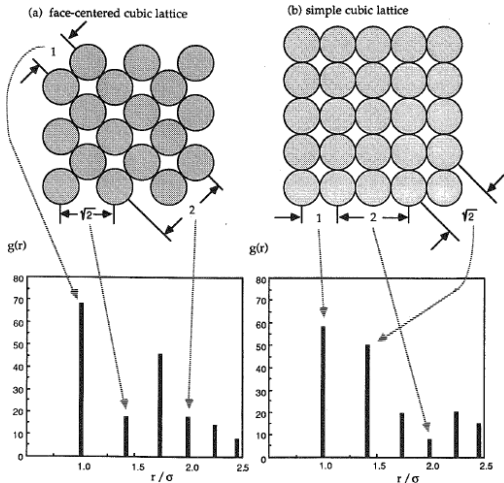
- the compressibility equation

$$kT \left( \frac{\partial \rho}{\partial p} \right) = 1 + \rho \int dr [g(r) - 1]$$



# Radial Distribution Function

RDF can be used to determine the characteristics of the atomic structure.



# Studying the system's properties

In molecular dynamics we compute the properties of interest by *calculating the time average* of those properties over the time of the simulation.

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n x(t)$$

In order to obtain a quality results we have to wait until the evolution of our simulation will take the studied system into “every” accessible point in the phase space. At that point we can say that the set of values  $x(t)$  that we use in the averaging is *truly a random set*.

# Studying the system's properties

Unfortunately, in order to make the system (non-periodic system) “probe” the entire phase space, we would have to wait *infinitely long time*.

Since this is too long for most people, we just have to satisfy our selves “*long enough*”.

## QUESTION

However, how long is that “long enough”?

This question can be answered with the help of statistics, by measuring the *correlations* of the property of interest.

## DEFINITION

It is clear that, the nearby points on a trajectory are *not independent*. Rather they are casually connected via the algorithm used to solve the equations of motion. Neighbouring are said to be *serially correlated*. Thus we can define correlation coefficient as:

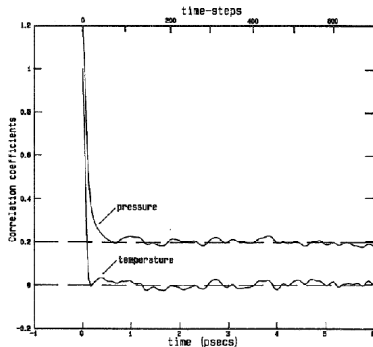
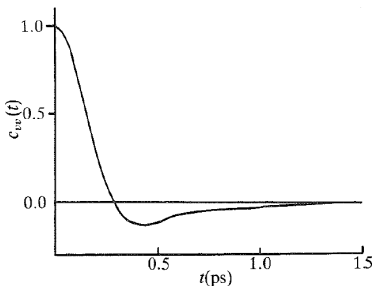
$$C(t) = \frac{\sum_k^M (x(t_k) - \langle x \rangle) (x(t_k + t) - \langle x \rangle)}{\sum_k^M (x(t_k) - \langle x \rangle)^2}$$

The coefficient measures how the value of  $x$  at time  $t_k + t$  is correlated to the value at earlier time  $t_k$ . In other words *“what is the influence of the earlier state of the system on its present state”*.

However these values are the same variable measured in different points in time. Thus,  $C(t)$  is usually called the *auto-correlation function*  $C_{xx}(t)$ .

# Correlation functions examples

At the start of the simulation  $C(t = t_0) = 1$ . Further the time average is subtracted from each  $x(t)$ , forcing  $C(t) \rightarrow 0$  when the elapsed time is “*long enough*”.



Computing a correlation function enables us to identify the *relaxation time*, a time, beyond which, the serial correlations are disrupted.

THANK YOU FOR YOUR ATTENTION.