

Title to find ...

Gautier VAILLANT

August 4, 2015

Acknowledgements

I would first like to gratefully thank Helmut Grubmüller and Bert De Groot for welcoming me in their lab. I also thank Carsten Kutzner who supervised my Internship. I also would like to Thank Bartosz Kohnke and Thomas Ullmann for providing me advice on my project.

Finally I would also like to thank Ivo Kabadschow and Andreas Beckmann who welcomed me in the Jülich Forschungszentrum to teach me the details of the FMM method.

Abstract

Simulating large pairwise interactions is a very important issue for Scientific research. It plays an important role in Astrophysics to know the dynamics of galaxies, in plasma physics or in our case in biophysics. This kind of simulations is typically with a complexity of $\mathcal{O}(N^2)$ which scales badly with the size of the system.

Some other techniques, such as the PME (Particle Mesh Ewald) and the FMM (Fast Multipole Method) are able to obtain a complexity of respectively $\mathcal{O}(N \log(N))$ and $\mathcal{O}(N)$.

La simulation de larges systemes de particules en interaction est tres importante pour le calcul scientifique. Elles jouent un role important en Astrophysique pour connaitre la dynamique des galaxies, en physique des plasmas ainsi que, dans notre cas, en Biophysique.

Contents

1	Methods for computing electrostatic forces	8
1.1	$\mathcal{O}(N^2)$ method	8
1.1.1	Naive Method	8
1.1.2	Possible improvements	9
1.2	Fourier Transform-Based methods	10
1.2.1	Ewald Summation	10
1.2.2	PME	12
1.3	Fast Summation methods	13
1.3.1	Mathematical preliminaries	14
1.3.2	Workflow of the algorithm	15
2	Comparing FMM and PME accuracy	17
2.1	Presentation of GROMACS	17
2.1.1	Structure of a File	17
2.1.2	PME parameters	18
2.2	Error plots	18
	Appendices	19
A	Structure of Gromacs files	20

Presentation of the Lab

Context of the Internship

The Context of this Internship is driven by the "SPPEXA (Software for exascale computing) / GromEx" project funded by the DFG (Deutsche Forschungsgemeinschaft). The Idea of this project is to create a flexible and fast solver for computing forces and potentials, which is a preliminary for molecular simulations.

A poster¹ of the project can be found below :

¹from <http://www.mpibpc.mpg.de/grubmueller/sppexa>

Contents



Figure 1: Poster for the SPEXXA project

Currently the method used for computing electrostatic forces is called the PME (Particle Mesh Ewald). It works nicely but one of its problems is that the algorithms cannot be efficiently parallelised tasks as there is a lot of communication between the CPU and GPU cores. The idea would be to replace this method with a new method called the Fast-Multipole Method which is based on a tree Structure and may allow an greater parallelization of the system as wanted.

So the Idea of the Internship is first to know if the both method method have the same accuracy and for which parameters.

Chapter 1

Methods for computing electrostatic forces

1.1 $\mathcal{O}(N^2)$ method

In this section, we will explain the most basic method to compute pairwise interactions and explain why the method leads to long computation times and sometimes artifacts.

1.1.1 Naive Method

The coulombic interaction between two charged particles can be written the following way:

$$\vec{F}_{A \rightarrow B} = \frac{q_A q_B \hat{r}_{AB}}{4\pi\epsilon_0 |R_{AB}|^2} \quad (1.1)$$

where q_A and q_B are respectively the charges of A and B, and R_{AB} is the distance between A and B.

In the thesis we will simplify the units of (1.1) for computational reasons by just writing :

$$\vec{F}_{A \rightarrow B} = \frac{q_A q_B \hat{r}_{AB}}{|R_{AB}|^2} \quad (1.2)$$

The Corresponding potential for a charged particle, with a charge q is :

$$V = \frac{1}{r} \quad (1.3)$$

1.1. $\mathcal{O}(N^2)$ method

The first, naive way to compute electrostatic forces is the following : in order to compute the force acting on one particle, it is needed to obtain the coulombic interaction for each pair of particles.

So if we consider a set of N charged particles, $N - 1$ interactions are needed to compute the force acting on one specific particle. So in order to know the forces of the set of particles, $N \cdot (N - 1)$ operations are needed, hence an algorithmic complexity of $\mathcal{O}(N^2)$.

This gives the following algorithm:

```
input : A set of  $N$  charged Particles
output: A List of the forces for each particle

For each particle  $i$ ;
for  $i \leftarrow 1$  to  $N - 1$  do
    | add interaction between particle  $i$  and particle  $j$  ;
    | for  $j \leftarrow i + 1$  to  $N$  do
    | | force  $[i] \leftarrow$  force  $[i] + \text{computeForce}(i, j)$  ;
    | end
end
```

Algorithm 1: Naive method

The complexity of such a computation limits its use to rather small systems and is not really usable for bigger systems such as proteins or astrophysical systems.

1.1.2 Possible improvements

A possible improvement is to limit the interaction to a certain radius : if the distance between two particles is greater than R_0 , then the force is set to 0.

So we have the following system :

$$\vec{F}_{A \rightarrow B} = \begin{cases} \frac{q_A q_B \hat{r}_{AB}}{|R_{AB}|^2} & \text{if } R_{AB} < R_0 \\ \vec{0} & \text{otherwise} \end{cases} \quad (1.4)$$

This technique is for example used for Lennard-Jones potentials ($V_{LJ} = 4\epsilon[(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^6]$), where the Intensity of the force is quickly decreasing. It allows to limit the number of interactions to only the close neighbours.

However, one of the problems of this optimisation, especially for long-range interactions such as coulombic interactions is that using a cut-off can lead to artefacts : A particle feels the force, then crosses the cut-off radius. Suddenly, the particle doesn't feel any force anymore, thus the artefacts as it is showed figure 1.1 .

1.2. Fourier Transform-Based methods

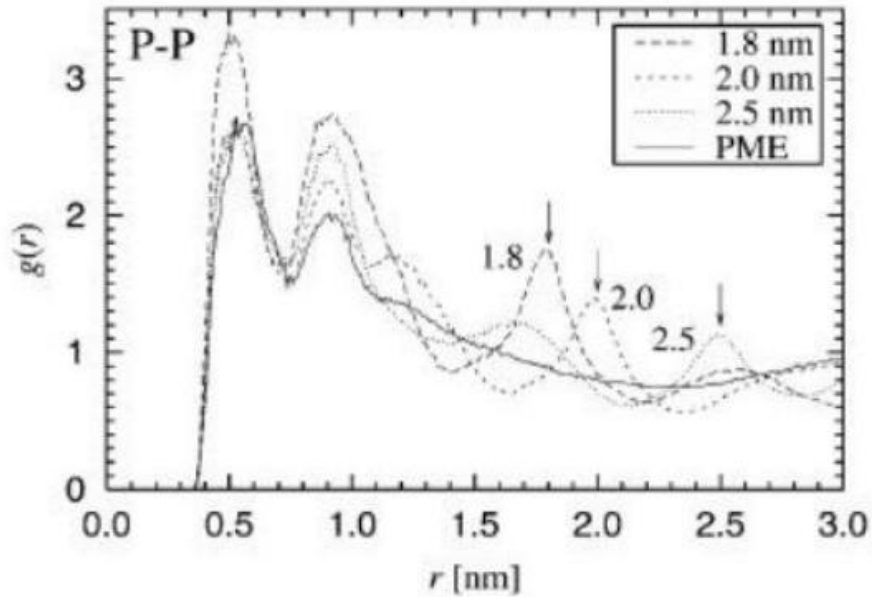


Figure 1.1: Radial distribution function (RDF) $g(r)$ between the two central atoms in the headgroup of a molecule: Cutoff distances are indicated by arrows. from ...

As we can see in figure 1.1, the radial distribution of the distance between two atom shows a peak, corresponding to the cutoff of the system. This shows that by using a cut-off technique we might see some artefacts.

So the two reasons we don't use a $\mathcal{O}(N^2)$ method is first because of its complexity, and using some optimization techniques can also lead to artifacts.

1.2 Fourier Transform-Based methods

In this section, we will explain techniques using periodic boundary conditions and Fourier transformation in order to compute the potentials and the forces of the particles.

1.2.1 Ewald Summation

This subset of techniques comes from a theoretical physics technique called the Ewald summation.

Using periodic boundary conditions, the potential V of one particle of the system is:

1.2. Fourier Transform-Based methods

$$V = \sum_{n_x, n_y, n_z} \sum_i^N \sum_j^N \frac{q_i q_j}{r_{ij}} \quad (1.5)$$

where n_x, n_y, n_z are the box index vector.

The equation 1.5 is conditionally convergent and slow to converge. One technique discovered by Ewald is to split the potential in two absolutely convergent terms and one constant term:

If the system is neutral, ie. $\sum_{i=1}^N q_i = 0$, The idea is to split the potential the following way :

$$V = \frac{1}{r} = \frac{f(x)}{r} + \frac{1-f(x)}{r} \quad (1.6)$$

$$V = V_{\text{direct}} + V_{\text{reciprocal}} \quad (1.7)$$

We want to choose f , so that $\frac{f(x)}{r}$ is quickly decaying and $\frac{1-f(x)}{r}$ is as smooth as possible. In this decomposition $\frac{f(x)}{r}$ is as small as possible, even 0 above a certain cut-off. Then, $\frac{1-f(x)}{r}$ is smooth enough so we can compute its Fourier transform with just a few \vec{k} vectors. This gives a quick computation of the reciprocal space.

a good choice is often $f(x) = \text{erfc}(\alpha x)$, where α is called the splitting parameter, and $\text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{+\infty} e^{-t^2} dt$

so we obtain the following potentials :

For the direct space :

$$V_{\text{direct}} = \frac{1}{2} \sum_{n_x, n_y, n_z} \sum_{i,j}^N q_i q_j \frac{\text{erfc}(\alpha r)}{r} \quad (1.8)$$

and for the reciprocal space :

$$V_{\text{reciprocal}} = \frac{1}{2\pi V} \sum_{i,j}^N q_i q_j \sum_{n_x^*, n_y^*, n_z^*} \frac{\exp -(\pi \vec{m}/\alpha)^2 + 2\pi i \vec{m} \cdot (r_i - r_j)}{m^2} \quad (1.9)$$

where n_x^*, n_y^*, n_z^* are the vectors in the reciprocal space.

1.2. Fourier Transform-Based methods

Once the electrostatic potentials are obtained, it is possible to differentiate the potentials to obtain the force on each particle.

let $p = (x, y, z)$ a coordinate of the system, then :

$$\vec{F}_p^{\text{direct}} = \vec{\nabla}_p V_{\text{direct}} \quad (1.10)$$

so we have:

$$\vec{F}_p^{\text{direct}} = q_i \sum_{i=1, i \neq j}^N \sum_{\mathbf{n}} q_j \frac{(r_{ij,n})_p}{r_{ij,n}^3} \left\{ \text{erfc}(\alpha r_{ij,n}) + \frac{2\alpha}{\sqrt{\pi}} r_{ij,n} \exp(-(\alpha r_{ij,n})^2) \right\} \quad (1.11)$$

$$\vec{F}_p^{\text{reciprocal}} = \frac{2q_i}{L} \sum_{i=1, i \neq j}^N \sum_{\mathbf{n}^* \neq 0} \frac{n_p^*}{n^{*2}} \exp\left(-\left(\frac{\pi m}{\alpha L}\right)^2\right) \sin \frac{2\pi}{L} m \cdot r_{ij} \quad (1.12)$$

Equations 1.11 and 1.12 can then be used to compute the force on each particle by adding the direct space and the reciprocal space contribution.

1.2.2 PME

One way to improve this method is to compute the fourier sum using the FFT (Fast Fourier Transform). This allows the algorithm to get a complexity of $\mathcal{O}(N \log N)$. There is different methods that are based on the Ewald summation, namely the **P3M** (Particle-Particle Particle-Mesh Method) or the **FFP** (Fast fourier poisson method).

In our case, we will focus on the **PME** as it is the method currently used in GROMACS.

We remind that the potential for the reciprocal space $V_{\text{reciprocal}}$ is written in equation (1.9):

$$V_{\text{reciprocal}} = \frac{1}{2\pi V} \sum_{i,j} q_i q_j \sum_{\mathbf{n}^*} \frac{\exp\left[-(\pi \vec{m}/\alpha)^2 + 2\pi i \vec{m} \cdot (r_i - r_j)\right]}{m^2}$$

This equation can be rewritten as :

1.3. Fast Summation methods

$$V_{\text{reciprocal}} = \frac{1}{2\pi V} \sum_{\mathbf{n}^* \neq 0}^N \frac{\exp -(\pi \vec{m}/\alpha)^2}{m^2} S(-\mathbf{n}^*) S(\mathbf{n}^*)$$

where $S(\mathbf{n}^*)$ is defined as the Structure factor:

$$S(\mathbf{n}^*) = \sum_{k=1}^N q_k \exp(2\pi i \mathbf{n}^* \cdot \mathbf{r}) \quad (1.13)$$

The idea of the method is the following : In order to apply a FFT to the system, it is needed to "map" the positions of the charges on a grid of size $p \times p$. We will define p as the order of the PME scheme. Let also define $\mathcal{F}(Q)$ the 3D FFT of Q . The charges q_i are mapped to the grid using interpolation. Originally, lagrange interpolation was used, but now a b-spline interpolation is used.

then the Structure factor can be approximated as its FFT :

$$S(\mathbf{m}) \approx \tilde{S}(\mathbf{m}) = \mathcal{F}(Q)(\mathbf{m}) \quad (1.14)$$

so the reciprocal energy can also be approximated by:

$$V_{\text{reciprocal}} \approx \tilde{V}_{\text{reciprocal}} = \frac{1}{2\pi V} \sum_{\mathbf{n}^* \neq 0}^N \frac{\exp -(\pi \vec{m}/\alpha)^2}{m^2} \mathcal{F}(Q)(\mathbf{m}) \mathcal{F}(Q)(-\mathbf{m}) \quad (1.15)$$

It can be shown that the complexity of such a system is $\mathcal{O}(N \log(N))$ which is a much bigger improvement compared to the $\mathcal{O}(N^2)$ complexity of the direct algorithm. However, one of its disadvantages is that it relies on a periodic sum, which is theoretically infinite.

1.3 Fast Summation methods

In the previous section we showed a method which allows to improve the speed of the computation. In this section, another algorithm will be explained, which has a complexity of $\mathcal{O}(N)$.

1.3. Fast Summation methods

1.3.1 Mathematical preliminaries

Let's move back to the potential created by a charged particle as in (1.3),

$$V = \frac{1}{d}$$

Let two particles $A(a, \alpha, \beta)$ and $B(r, \theta, \phi)$, separated by a certain distance d : The distance between the particles is $d = |a - r|$. We would like to achieve is to "factorize the addition", having a product of one function depending only on a and one depending only on r .

The inverse distance can be written as :

$$\frac{1}{d} = \frac{1}{|a - r|} = \frac{1}{\sqrt{a^2 + r^2}}$$

This distance can then written as a series :

$$\frac{1}{d} = \frac{1}{\sqrt{a^2 + r^2}} = \sum_{l=0}^{+\infty} P_l(u) \mu^l \quad (1.16)$$

where $\mu = \frac{a}{r}$ and $u = \cos \gamma$. The $P_l(u)$ are the Legendre polynomials of degree l .

with some manipulation as explained [ADD REFERENCE], we obtain :

$$\frac{1}{|r - a|} = \sum_{l=0}^{+\infty} \sum_{m=-l}^{+l} \frac{(l-m)!}{(l+m)!} \frac{a^l}{r^{l+1}} P_{lm} \cos(\alpha) P_{lm} \cos(\theta) e^{-im(\beta-\alpha)} \quad (1.17)$$

in order to approximate the scheme, we can truncate the series to a certain order p , we call this order the **multipole order**

$$\frac{1}{|r - a|} \simeq \sum_{l=0}^{\overset{p}{+}} \sum_{m=-l}^{+l} \frac{(l-m)!}{(l+m)!} \frac{a^l}{r^{l+1}} P_{lm} \cos(\alpha) P_{lm} \cos(\theta) e^{-im(\beta-\alpha)} \quad (1.18)$$

We can now rewrite the summation the following way :

$$\frac{1}{|r - a|} \simeq \sum_{l=0}^p \sum_{m=-l}^{+l} \underbrace{\frac{a^l}{(l+m)!} P_{lm}(\cos(\alpha)) e^{-im\beta}}_{O_{lm}(\mathbf{a})} \underbrace{\frac{(l-m)!}{r^{l+1}} P_{lm}(\cos(\theta)) e^{+im\phi}}_{M_{lm}(\mathbf{r})} \quad (1.19)$$

hence,

1.3. Fast Summation methods

$$\frac{1}{|r - a|} \simeq \sum_{l=0}^p \sum_{m=-l}^{+l} O_{lm}(\mathbf{a}) M_{lm}(\mathbf{r}) \quad (1.20)$$

We have shown that it is possible to "factorize" the inverse of the distance $\frac{1}{d}$ so we can obtain two independent terms for two independent particles $O_{lm}(\mathbf{a})$ and $M_{lm}(\mathbf{r})$

We can then define the multipole moment $\omega_{lm}(q, \mathbf{a}) = qO_{lm}(\mathbf{a})$ and the Taylor-like moment $\mu_{lm}(q, \mathbf{r}) = qM_{lm}(\mathbf{r})$

Then, it is possible to show that it is possible to obtain the following "bipolar expansion" for two particles associated with two different origins :

$$\frac{1}{|\mathbf{a}_1 - \mathbf{a}_2 + \mathbf{R}|} = \sum_{l=0}^{+\infty} \sum_{j=0}^{+\infty} \sum_{m=-l}^{+l} \sum_{k=-j}^{+j} (-1)^j \cdot O_{lm}(\mathbf{a}_1) \cdot M_{l+j, m+k}(\mathbf{R}) \cdot O_{jk}(\mathbf{a}_2) \quad (1.21)$$

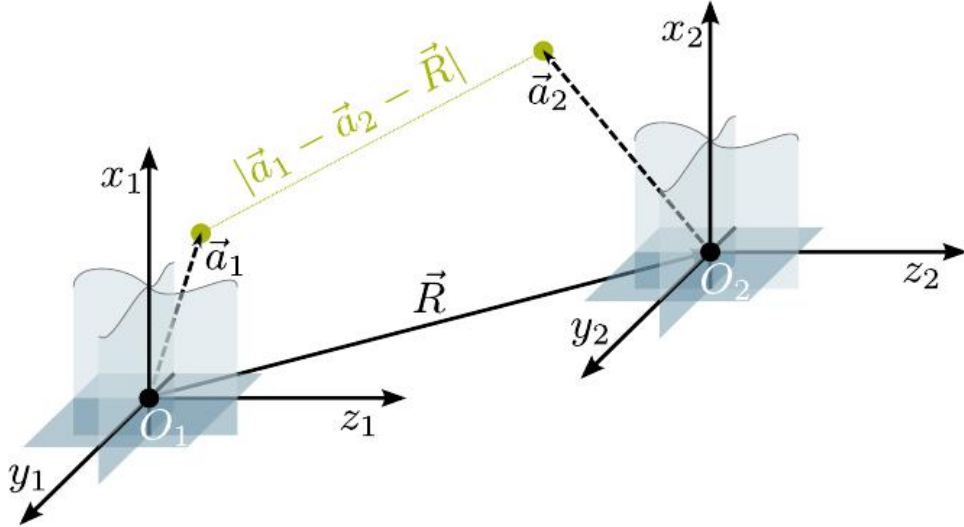


Figure 1.2: Scheme of two particles expanded according to two different origins [from ADD REFERENCE]

1.3.2 Workflow of the algorithm

Once some Mathematical preliminaries are set, it is possible to explain the workflow of the Algorithm.

1.3. Fast Summation methods

Splitting the Space

The first step of the scheme is to split the space in order to generate different groups. The idea is to recursively split the space in eight octant, created the so-called structure of an *octree*.

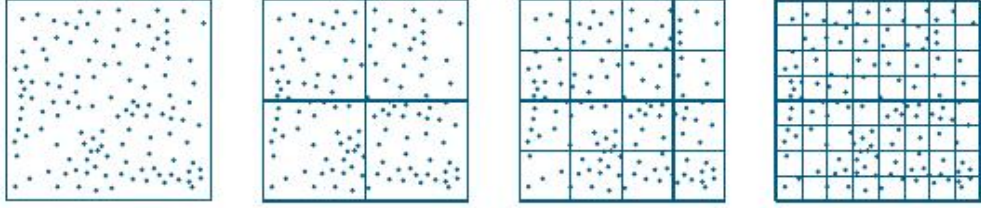


Figure 1.3: Example of a division of the space in boxes. The following example is in 2D so the structure of the space is a *quadtrees*.

The number of subdivisions is called the *depth* D . For example, in figure 1.3, we can observe from left to right the depths 0, 1, 2 and 3.

It is then possible to assign each particle of the simulation to the boxes of lowest depth.

Computing the Multipole moments

Once every particle is assigned to its box, we will compute the multipole moment ω_{lm}^j of each box.

$$\omega_{lm}^j(q, \mathbf{a}) = q^j a_j^l \tilde{P}_{lm}(\cos(\alpha_j)) e^{-im\beta_j} \quad (1.22)$$

Then each multipole order can be moved to the Center of the box using the so-called $M2M$ (Multipole to Multipole) operator:

$$\omega(\mathbf{a} + \mathbf{b}) = \sum_{j=0}^k \sum_{k=-j}^j \omega_{jk}(\mathbf{a}) O_{l-j, m-k}(\mathbf{b}) \quad (1.23)$$

$O_{l-j, m-k}(\mathbf{b})$ is called the $M2M$ operator

Chapter 2

Comparing FMM and PME accuracy

In this chapter I will more precisely explain my work done in the Lab, which consisted in comparing the PME methods, which is for example used by gromacs, and the FMM method, used by a solver made between the Jülich Forschungszentrum and the Max Planck Institute for Biophysics in Göttingen.

I will first explain for Gromacs work and then

2.1 Presentation of GROMACS

GROMACS (GRONingen MACHINE for Chemical Simulations) is a simulation software used for Molecular dynamics. It was originally developed in the Biophysical Chemistry department of University of Groningen but it is now developed around the world. GROMACS is made to be as fast as possible using all possible techniques to improve its performance (MPI, GPU computing)

2.1.1 Structure of a File

The program is used by modifying text files that giving information on the structure of the system that has to be simulated and the parameters of the simulation. The files are the following

- The *.pdb file are the *Protein DataBank* file : In this file is denoted the position and the type of the particles of the system. It also gives the size of the simulation box

2.2. Error plots

- The *.top files (namelt *topology* files) are the file where the properities of the atoms are defined. These properties are for example the charge, the Van-der-Waals parameters or the binding forces of the system. This configuration is often done with force field files such as AMBER or CHARMM
- The *.mdp files are defining the physical and computational parameters of the simulation. It is defining the methods used for the simulation. In our case we are mostly interested in the electrostatics part of the computation ; it is then possible to select the method (PME or Cut-Off methods) and the parameters for the PME

2.1.2 PME parameters

2.2 Error plots

Appendices

Appendix A

Structure of Gromacs files

The contents...