Master Thesis

Numerical Study of Modified Newtonian Dynamics

Author:

Huda Nasrulloh

1815011064

Supervisor:

Prof. Masato Kimura

Thesis submitted for the degree of

Master in Computational Science

Kanazawa University

Graduate School of Natural Science and Technology

Division of Mathematical and Physical Sciences

July 2020

This page is intentionally left blank.

Acknowledgements

First of all, I would like to be grateful to God, the Almighty, the Most Gracious and the Most Merciful who blesses me for entire of my life. Indeed, it is the only His favours that got me the ability to compose this thesis.

Basically, this thesis is a result of my academic and research works in Double Degree Master Program in Graduate School of Natural Science and Technology, Division of Mathematical and Physical Sciences at Kanazawa University for about one year. Surely, there were many people contributing in accomplishing this thesis. They are parents, lecturers, academic staffs in college, colleagues, supervisor, and many others that I cannot mention them all here.

I especially would like to express my deep gratitude to my supervisor, Professor Masato Kimura, for guiding me patiently on doing research and composing this thesis. Besides that, I greatly appreciate the academic environment in Kimura-Notsu Laboratory that helped me in struggling on this work, DDP students for sharing me something to make this research completely well done. Then, special thank should be given to my research partner, Alifian Mahardika, who always helped me do well.

Finally I also thank to Japan Student Services Organization (JASSO) for funding me study one year in Japan. I realized that without grant from JASSO scholarship, financial problems during study at Japanese university would be hard for me.

And He gave you from all you asked of Him. And if you should count the favor of Allah , you could not enumerate them. Indeed, mankind is [generally] most unjust and ungrateful.
(QS. Ibrahim : (14:34))

Dedicated to :

My family who pray me in their continuos prayers.

above every knower there is a higher Knower.

(QS. Yusuf : 76)

Abstract

In this work we simulate the particle motion as a astronomical object which is called galaxy. The particle we set as naturally condition in galaxy scale. Then we use two governing equation to provide this work. The first we implement by using Newtonian dynamics (ND) and another equation using Modified Newtonian dynamics (MOND). By comparing that two equations we get the information how successful from each equation can do the simulation. We consider also two numerical method which are Euler method and symplectic method. Based on all these works we can conclude that MOND $\cdots$.

Key words: *MOND, Newtonian dynamic, Euler method, symplectic method*

Contents

List of figures

List of tables

Frequently used abbreviations

Chapter 1

Introduction

Where are we actually exist ? Some of us might answer the location of each sweet home. It's the same if we observe from an astrophysical point of view, we are placed on earth, in the Milky Way galaxy with fixed coordinates. Milky Way is actually just one of the names of galaxies that can be observed for astronomical observations. There are so many galaxies in our universe, and we still don't really know the all of information. This is a great mystery in our universe as an astrophysicist.

One of interesting issue from galaxies in addition to find the latest astronomical objects is information about the content and dynamics of the galaxy. The contents of galaxies are usually studied in particle physics. Where every single particles have an unique behavior. But about galaxy dynamics we can observe them as mathematical view. Therefore, the main problem that will be given more explanation in this thesis, we study numerical solution of galaxies.

Basically, we try to consider all the physical conditions that give responsibility for this case. We might combine the two methods between astrophysical theory and mathematical models as best we can get physical meaning and also best illustrations. For astrophysical methods we consider Modified Newtonian's Dynamic (MOND) as a theory alternative to explain the dynamics of galaxies. In addition, MOND actually doesn't really consider galactic content, which means dark matter. So that would be a good way for us to go through this section so that it only functions in mathematical simulations.

1.1 Theory of Galaxy Dynamics

An astronomical object called a galaxy, is a large ensemble of stars and other material that orbits about a shared center and its constituents are united by mutual gravitational interactions. Galaxies come in various global forms and internal morphology. In this section we will be given an overview of galaxies for mathematical modeling aid numerical simulation of the dynamics of the galaxy.

1.1.1 General Review of Galaxy

Galaxies are star-bound systems of gravity, remnants of stars, interstellar gas, dust, and dark matter [1]. The galaxy always moves dynamically. So, by this simulation we can understand past, present and future predictions of galaxies.

We know that each galaxy has differences form one and the others. It's easy to identify the type of galaxy we are looking at. We are familiar with galaxies: ellipses, spiral and disks [2]. One of the best known examples is our Milky Way galaxy as an example of spiral galaxy.

In our galaxy, the Milky Way, has many kinds of matter, gas and dust inside it. Observations provide information that our galaxy has weight about 1.5 trillion solar mas. This part of around 200 billion stars, including 4 million solar masses from black holes at the center of galaxies.

The other information about our Milky way is the galactic diameter around 10^5 light-year. Another scale might sometimes use galactic diameter in Kpc scale, where 10^5 ly ≈ 30.7 Kpc. In addition, the speed of the star to orbit the center of the galaxy has about ~ 230 km/sec.

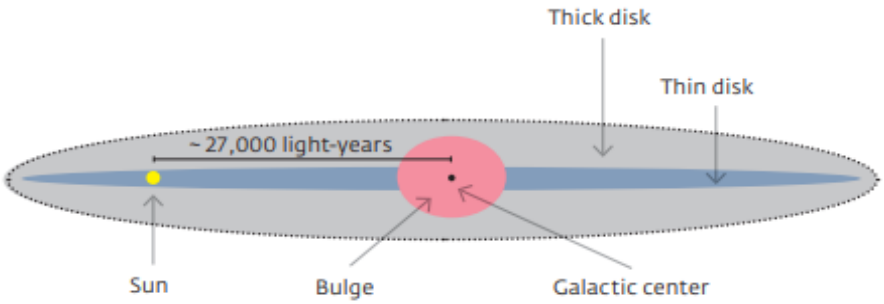

Figure 1.1: Part of Milky way galaxy [3]

Fig. 1.1 shows that our Milky Way galaxy has a complex structure. Each component has a different behavior. As in the center of the galaxy, it remains in the life of the galaxy, which can one day destroy all matter as an active black hole inside that. Then, galactic disk as a medium of many star that make a circular motion. Next for halo galaxies which have the task of guarding globular galaxies. Galaxy content won't come out because of the Halo effect.

Table 1.1: Table of Galaxy properties

Variable	Physical Meaning	Unity	Typical Value
M_{tot}	Total mass of galaxy	$[M_\odot]$	$\sim 1 \times 10^{12}$ [3]
N	Number of stars	-	$\sim 2 \times 10^{11}$ [4]
m_i	Average of each star mass	M_{tot}/N	$\sim 10 M_\odot$ [5]
R	Radius	[Kpc]	~ 15
$\|v\|$	speed of star	[km/s]	230
ω	Angular elocity	$\sim 1/\sqrt{R^3}$	0.017 [Kpc]
T	Period of galaxy rotation	galaxy-year	1

The sun as the center of the solar system, the place where we are, has a distance from the center of about 27.000 = 8.4 Kpc. The elegance of astronomical objects will be studied by making numerical simulations. We don't consider dark matter (DM), so we can ignore that effect by modifying Newton's dynamics. Next we will give a brief discussion as background of this research.

1.2 Possibility to solve as Dark Matter Problem

As we discussed before, the most important problem of discrepancy between the observation and Newton's law prediction is impacted by strange matter. The Scientist familiar to say as dark matter (DM). This decade, Scientist have been searching for what actually information about DM. The reason for this persistence is that dark matter is needed to account for the fact that galaxies don't seem to obey the fundamental laws of physics. However, dark matter searches have remained unsuccessful.

The mystery of what dark matter actually is still remaining as an ultimate challenge of modern particle physics. The core question is whether that indeed a missing mass source, such as a new type of matter, or whether the gravitational law is simply different at large scales. If we refers in the observation recently results based on Hubble Space telescope, which have substantially revised and improved understanding of the stellar mass function [6]. This mean that in galaxy scale, how is the distance as possible enough to observe it, also potentially to solve the discrepancy problem.

Literally, a whole reason in physical case actually indicate to condition that our universe need more enough matter as a content of this universe. So, basically the main concept about dark matter is not only for galaxy case but also in the cosmic scale case. We same as exactly know, if we consider the real condition such as dark matter problem, we need to more time to explain that. We can trough the realization if our work actually can precisely illustrating the general work between mathematical work and physical order. Of course it will be complicated enough to do that in same time. Therefore we don't really close our eyes about the possibility of dark matter problem. We just work in the best way to get as well as possible enough to predict and give best explanation.

1.3 Point of View MOND

The biggest mystery in cosmology and astrophysics is missing information about dark matter and energy. Two of them are inevitable when we study in large-scale physics, such as galaxies, the universe and physics larger scales. For this reason, a scientist named Milgrom proposed his hypothesis. He said that to solve the DM problem in large-scale, physics need to modify basic Newtonian equations [7].

Furthermore, by considering this motivation, some researchers can obtain good data when compared with observing data. As in the figure below, MOND predictions get similar data from galaxies, NGC 2403 observations The graph

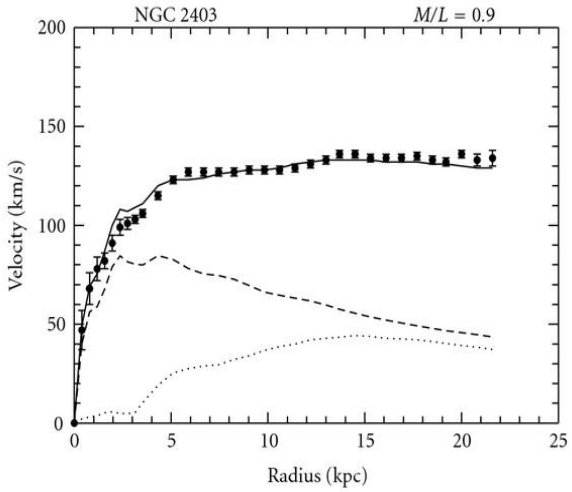

Figure 1.2: Graphic observationally data with show by point and MOND the solid line. Dash-line for Keplerian prediction, dot-line for visible matter other method.[8]

in fig(1.2) shown how successful MOND can predict velocity of the content galaxy. Meanwhile we can see that dash-line from the figure has been resulted from Newtonian dynamic as Keplerian prediction.

Basically all of the information from dynamics of galaxy is a big motivation why we did this research. We want to make a simulated galaxy, and make observations about the results obtained. So, based on all of explanation we will spread the written composition of this thesis. Chapter 2 we give an

introduction to Newton's dynamics as a theoretical basis that provides an explanation of particle motion. Then we also give a brief summary of the scaling of galaxies. This is very important because galaxy objects have large-scale entities where they can be troublesome in the program. In Chapter 3 Modified Newtonian Dynamics (MOND) is displayed to be the main topic of this thesis. Furthermore, the numerical scheme will be displayed in Chapter 4.2 including two methods, namely Euler and symplectic method. Finally, we show numerical results in Chapter 5 and really provide a brief discussion based on the results obtained.

Chapter 2

Newtonian Dynamics

2.1 Introduction

Newtonian dynamics is a mathematical equation that provides to predict the motions of various object which exist in the world around us. The implementation of this model has been familiar in the physical case, since for small scale interaction until large scale. In this chapter we give a brief explanation about Newtonian dynamics as a fundamental equation that we intent to modify and some of notations in Newton's law.

2.2 Governing Equation of Newtonian Dynamics

2.2.1 Acceleration in Newtonian Dynamics

We consider $X(t) = \{x_i(t)\}_{i=1}^{N} \subset \mathbb{R}^d$ which denotes the positions of N particles in $\mathbb{R}^d (d = 2 \text{ or } 3)$ at time t. We suppose that

$$x_i(t) \neq x_j(t) \qquad (i \neq j) \tag{2.1}$$

The velocity and acceleration of $x_i(t)$ are denoted by

$$\begin{cases} v_i(t) := \dot{x}_i(t), \\ a_i(t) := \ddot{x}_i(t), \end{cases}$$

Then, Newton's second law of motions is given as

$$m_i a_i(t) = F_i(t) \qquad (i = 1, 2, \ldots, N), \tag{2.2}$$

where miss a mass of the particles x_i and $F_i(t)$ is the total force acting on $x_i(t)$.

Newton's law of universal gravitation takes the form :

$$F_i(t) = GM \sum_{\substack{j=1 \\ (j \neq 1)}}^{N} \frac{m_j}{r_{ij}(t)^3} x_{ij}(t), \tag{2.3}$$

where G is the gravitational constant and we define $r_{ij}(t) := |x_j(t) - x_i(t)|$, $x_{ij}(t) := x_j(t) - x_i(t)$. If we substitute (2.2) into the last equation (2.3), so we get the acceleration of particle

$$\begin{aligned} m_i a_i(t) &= Gm_i \sum_{j=1, j \neq i}^{N} \frac{m_j}{r_{ij}(t)^3} x_{ij}(t) \\ a_i(t) &= G \sum_{j=1, j \neq i}^{N} \frac{m_j}{r_{ij}(t)^3} x_{ij}(t). \end{aligned} \tag{2.4}$$

We define

$$f_i(X) := G \sum_{\substack{j=1 \\ (j \neq 1)}}^{N} \frac{m_j}{|x_j - x_i|^3}(x_j - x_i). \qquad \left(X := \{x_j\}_{j=1}^{N}\right) \tag{2.5}$$

Then (2.4) becomes

$$\ddot{x}_i(t) = f_i\left(X(t)\right). \tag{2.6}$$

To avoid the singularity at $x_i(t) = x_j(t)$, using a small regularization parameter $\epsilon > 0$ [9]. We often replace $r_{ij}(t)$ by $r_{ij}^{\epsilon}(t) = (r_{ij}(t)^2 + \epsilon^2)^{1/2}$ in the numerical simulation. So, the last equation becomes

$$a_i(t) = f_i^{\epsilon}(X(t)), \tag{2.7}$$

where $f_i^{\epsilon}(X(t)) := G\sum_{j=1,j\neq i}^{N} \frac{m_j}{(r_{ij}^{\epsilon})^3} x_{ij}(t)$. The last equation that we do simulation for ND case in the program.

2.3 Conservation of Energy

For a particle system $X(t) = \{x_i(t)\}_{i=1}^{N}$, we denote its kinetic energy by

$$E_k(t) := \sum_{i=1}^{N} \frac{1}{2} m_i |v_i(t)|^2; \tag{2.8}$$

and also its potential energy by

$$E_p(t) := \sum_{1\leq i<j\leq N} \frac{m_i m_j G}{|x_j(t) - x_i(t)|}. \tag{2.9}$$

We can prove energy conservation property for the Newtonian dynamics

Theorem 2.3.1 (Conservation of energy)**.** *If $X(t) = \{x_i(t)\}_{i=1}^{N}$ satisfies (2.1) and (2.4). Then $E_k(t) - E_p(t) = Const$ holds.*

Proof.

$$\begin{aligned} \frac{d}{dt}E_k(t) &= \frac{d}{dt}\left(\sum_{i=1}^{N}\frac{1}{2}m_i|v_i(t)|^2\right) \qquad (2.10)\\ &= \sum_{i=1}^{N} m_i v_i(t)\cdot \dot{v}_i(t)\\ &= \sum_{i=1}^{N} m_i \dot{x}_i(t)\cdot f_i(X(t)) \end{aligned} \qquad (2.11)$$

We define the potential energy as

$$\mathcal{E}_p(X) := \sum_{1\le i<j\le N}\frac{m_i m_j G}{|x_j - x_i|} \quad \left(X=\{x_i\}_{i=1}^{N}\right). \qquad (2.12)$$

Then, the gradient of $\mathcal{E}_p(X)$ which respect to a variable x_i where $(i = 1, 2, \ldots, N)$ is given by

$$\begin{aligned} \nabla_{x_i}\mathcal{E}_p(X) &= \nabla_{x_i}\left(\sum_{j=1,j\neq 1}^{N}\frac{m_i m_j G}{|x_j - x_i|}\right)\\ &= \sum_{j=1,j\neq i}^{N} m_i m_j G\frac{x_j - x_i}{|x_j - x_i|^3}\\ &= m_i f_i(X), \end{aligned}$$

using this relation, we have

$$\begin{aligned} \frac{d}{dt}E_p(t) &= \sum_{i=1}^{N} m_i\dot{x}_i(t)\cdot f_i(X(t))\\ &= \sum_{i=1}^{N}\dot{x}_i(t)\cdot\nabla_{x_i}\mathcal{E}_p(X(t))\\ &= \frac{d}{dt}\mathcal{E}_p(X(t)) = \frac{d}{dt}E_p(t). \end{aligned}$$

Hence, we obtain

$$\frac{d}{dt}\left(E_k(t) - E_p(t)\right) = 0. \qquad (2.13)$$

■

2.4 Rotating particle system

We consider uniformly located N particles $\{x_i(t)\}_{i=1}^N \subset \mathbb{R}^2$ with an additional fixed particle at origin $x_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ of mass $m_0 \geq 0$ (commonly as an extra large mass). We set $X(t) = \{x_i(t)\}_{i=0}^N$. We suppose that $m_0 > 0$ or $N \geq 2$, and $x_i(t)$ has the following form :

$$x_i(t) = R \begin{pmatrix} \cos\left(\omega t + \frac{2\pi j}{N}\right) \\ \sin\left(\omega t + \frac{2\pi j}{N}\right) \end{pmatrix} \qquad (i = 1, \ldots, N) \tag{2.14}$$
$$m_j = m \qquad (j = 1, \ldots, N),$$

as illustrated in figure 2.1,

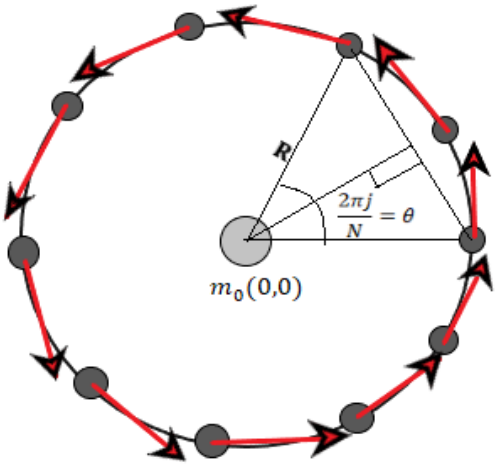

Figure 2.1: Rotating N particles and a fixed particle at origin.

We remark that $r_{ij}(t)$ is given by

$$r_{ij}(t) = 2R \sin \frac{\pi |j-i|}{N} \tag{2.15}$$

From 2.15 we have

$$\ddot{x}_i(t) = -\omega^2 x_i(t). \tag{2.16}$$

Then

$$f_i\left(X(t)\right) = mG \sum_{j=1, j\neq 1}^{N} \frac{x_j(t) - x_i(t)}{r_{ij}(t)^3} - m_0 G \frac{x_i(t)}{R^3} \qquad (i = 1, \ldots, N). \tag{2.17}$$

Then, from the symmetry it is enough to consider $t = 0$, $i = N$:

$$\begin{aligned}
f_N(0) &= mG \sum_{j=1}^{N-1} \frac{x_j(0) - x_N(0)}{r_{jN}(0)^3} - m_0 G \frac{x_N(0)}{R^3} \\
&= \begin{pmatrix} mG \sum_{j=1}^{N-1} \frac{R\left(\cos\frac{2\pi j}{N} - 1\right)}{8R^3 |\sin\frac{\pi j}{N}|^3} - m_0 G \frac{1}{R^2} \\ 0 \end{pmatrix} \\
&= -\frac{G}{R^3} \left(m \sum_{j=1}^{N-1} \frac{\sin^2 \frac{\pi j}{N}}{8|\sin\frac{\pi j}{N}|^3} + m_0 \right) \begin{pmatrix} R \\ 0 \end{pmatrix} \\
&= -\frac{1}{R^3} G \left(m \sum_{j=1}^{N-1} \frac{1}{8|\sin\frac{\pi j}{N}|} + m_0 \right) x_N(0) \\
&= -\frac{A}{R^3} x_N(0).
\end{aligned} \tag{2.18}$$

Where

$$A := G \left(m \sum_{j=1}^{N-1} \frac{1}{8|\sin\frac{\pi j}{N}|} + m_0 \right). \tag{2.19}$$

For general $t \geq 0$, and $i = 1, \ldots, N$, the equation becomes

$$f_i\left(X(t)\right) = -\frac{A}{R^3} x_i(t). \tag{2.20}$$

Hence we have

$$\ddot{x}_i(t) = -\frac{A}{R^3} x_i(t). \tag{2.21}$$

Comparing with (2.16) and (2.21), we obtain

$$\omega^2 R^3 = const = A \tag{2.22}$$

In particular, the speed of $x_i(t)$ is given by

$$|v_i(t)| = \omega R = \sqrt{\frac{A}{R}}. \tag{2.23}$$

Chapter 3

Modified Newtonian Dynamics

3.1 Introduction of MOND

Physics of the universe is one of elegant science that since have a many mystery inside them. So many kind of physicist work on there, to find all of mystery in our universe. One of the big mystery is about strange particle in our life. That matter we usually called as dark matter (DM). The word 'dark' is indicating what actual information about it still mystery.

Naturally, gravitational attraction of each cosmic objects is evidently higher than we expected in General Relativity or Newtonian dynamics from the distribution of the observable matter[6].

As far we know, the amount of DM in the Universe is much larger than that of the observable matter. There is evidence that DM cannot not arbitrary particle which formed by made of any known kind of particles.

These particles are not expected by the Standard model of particle physics. Dark matter has a great explanation of cosmic dynamic which the result about information large scale still covered as observational data, like (Planck Collaborations, Einstein et.al). Here we necessary to inform that the explanation about particle physics will not be given deeper, just for preliminary of our study case. So, the reader can be found this explanation in physical reference.

Also we still have the other way to solve the DM problem, that is to assume that we are able to detect most of the matter in the Universe. Then we need

to modify some of the standard laws of physics. These laws were derived from laboratory experiments and the Solar system observations. But on the galactic scales, many quantities (acceleration, angular momentum, mean density,etc) take values different from those in these experiments by many orders of magnitude.

Caused these conditions we choose the second way to modified from established equation. Newtonian second law or usually called Newtonian dynamic we modified s.t. which we called Modified Newtonian Dynamics (MOND). From this modified we want to show some of differentiate of this approximation beside we also need it to solving the rotational simulation of the galaxy.

3.2 Governing Equation of MOND

In physics, MOND (Modified Newtonian Dynamics) has actually been another way to describe dynamical of all about interaction can't explain by Newton's law, as well. Milgrom, the name who has propose equation, give adding the small correction in the order of accelerating of the object. He propose the equation caused to answer ND which can't give predict well from galaxy observation.

Basically, from the big reason why MOND should be proposed in physics large scale, we will try to show the formulating of MOND equation. It is mean that we should give the basic modified of ND. Therefore we show like bellow

$$F = m\mu\left(\frac{|a|}{\alpha_0}\right)a. \tag{3.1}$$

We introduced $\alpha_0 > 0$ as critical acceleration ($\alpha_0 \sim 10^{-8}cm/s^2 \approx 1.2 \times 10^{-10}km/s^2$). Then $\mu(\cdot)$ as function of interpolating this equation. We defined $s := \frac{|a|}{\alpha_0}$ for simplify our formula. Furthermore, we choose $\mu(s)$ for all possible $(0 \leq s \leq \infty)$, it is indicated a non-decreasing function with

$$\mu(0) = 0$$

and also

$$\lim_{s\to\infty} \mu(s) = 1$$

. Specifically for our case we best fixed the value of parameter $\mu(s)$ as follow this equation

$$\mu(s) = \frac{s}{\sqrt{1+s^2}}. \tag{3.2}$$

As we know that s we defined as $s := \frac{|a|}{\alpha_0}$.

Furthermore, start from this information and equation (3.1) that we give proposition like followed this

Proposition 3.2.1.

We suppose that $\mu(s)$ is given by (3.2), then (3.1) is equivalent to

$$F\beta\left(q\right) = ma \tag{3.3}$$

where $q := \frac{|f|}{\alpha_0}$ and $\beta(q) := \sqrt{\frac{1}{q}\varphi^{-1}(q)}$

Proof. We define $F := mf$, then equation (3.1) can rewrite as

$$m\mu\left(\frac{|a|}{\alpha_0}\right)a = mf \tag{3.4}$$

$$\mu\left(\frac{|a|}{\alpha_0}\right)a = f \tag{3.5}$$

$$\mu\left(\frac{|a|}{\alpha_0}\right)\frac{|a|}{\alpha_0} = \frac{|f|}{\alpha_0} \tag{3.6}$$

where $s := \frac{|a|}{\alpha_0}$ and $\varphi(s) := s\mu(s)$. So, from equation (3.6) we have

$$s\mu(s) = \frac{|f|}{\alpha_0} \quad \rightarrow \quad \varphi(s) = \frac{|f|}{\alpha_0}. \tag{3.7}$$

If $\mu(s) = \frac{s}{\sqrt{1+s^2}}$, then $\varphi(s) = \frac{s^2}{\sqrt{1+s^2}}$. Therefore we can rewrite equation (3.7) as

$$\varphi(s) = \frac{s^2}{\sqrt{1+s^2}} \quad \rightarrow \quad q = \frac{s^2}{\sqrt{1+s^2}} \tag{3.8}$$

Thus we can do this calculation

$$\begin{aligned} q^2 &= \frac{s^4}{1+s^2} \qquad (3.9)\\ s^4 - q^2 s^2 - q^2 &= 0 \\ s^2 &= \frac{q^2 + \sqrt{q^4+4q^2}}{2}. \\ s &= \sqrt{\frac{q^2 + \sqrt{q^4+4q^2}}{2}}. \qquad (3.10) \end{aligned}$$

By using relation between $\varphi(s)$ and s, we have the equation satisfies

$$\varphi^{-1}(q) = \sqrt{\frac{q^2 + \sqrt{q^4+4q^2}}{2}}, \qquad (3.11)$$

for $\beta(q)$ which also satisfies

$$\beta(q) = \sqrt{\frac{1}{q}\varphi^{-1}(q)} \quad \rightarrow \quad \beta(q) = \sqrt{\frac{1+\sqrt{1+\frac{4}{q^2}}}{2}}. \qquad (3.12)$$

Furthermore, equation (3.7) also can rewrite as

$$\begin{aligned} s &= \varphi^{-1}\left(\frac{|f|}{\alpha_0}\right) \\ \frac{|a|}{\alpha_0} &= \varphi^{-1}\left(\frac{|f|}{\alpha_0}\right) \\ |a| &= \alpha_0 \varphi^{-1}\left(\frac{|f|}{\alpha_0}\right) \\ a &= \alpha_0 \varphi^{-1}\left(\frac{|f|}{\alpha_0}\right)\frac{f}{|f|} \\ a &= \frac{1}{q}\varphi^{-1}(q) f \\ a &= \beta(q) f \qquad (3.13) \end{aligned}$$

If each sides we multiply by m, we obtain the result as a prove of this propo-

sition

$$ma = \beta(q)mf$$
$$ma = \beta(q)F \tag{3.14}$$

■

3.2.1 Rotating Particle System for MOND

In the previous section, we have derived symmetry particle for ND case. Then, this section we will show the symmetry particle of MOND. Overall, for MOND also have the same way to solve this problem. So we can refer into ND step as consider uniformly located N particles with particle fixed at origin. For detail step we can follow start from equation (2.15) until obtain equation (2.19).

Next we recall equation (3.6) in general form

$$\mu\left(\frac{|\ddot{x}_i(t)|}{\alpha_0}\right)\ddot{x}_i(t) \;=\; f_i\left(X(t)\right), \tag{3.15}$$

where $f_i\left(X(t)\right)$ is given by (2.20). Therefore, we have acceleration for rotating system in MOND as

$$\begin{aligned}\ddot{x}_i(t) &= \beta\left(\frac{|f_i\left(X(t)\right)|}{\alpha_0}\right) f_i\left(X(t)\right) \\ &= -\frac{A}{R^3}\beta\left(\frac{A}{\alpha_0 R^2}\right) x_i(t),\end{aligned} \tag{3.16}$$

for $|f_i\left(X(t)\right)|$ which satisfies

$$|f_i\left(X(t)\right)| \;=\; \left|\frac{A}{R^3}x_i(t)\right| \tag{3.17}$$
$$\;=\; \frac{A}{R^2}. \tag{3.18}$$

Similar with ND, we comparing (2.16) and (3.16), we have

$$\omega^2 = \frac{A}{R^3}\beta\left(\frac{A}{\alpha_0 R^2}\right) \tag{3.19}$$

$$\omega = \sqrt{\frac{A}{R^3}\beta\left(\frac{A}{\alpha_0 R^2}\right)}, \tag{3.20}$$

and speed has a result

$$\begin{aligned} |v_i(t)| &= \omega R \\ &= R\sqrt{\frac{A}{R^3}\beta\left(\frac{A}{\alpha_0 R^2}\right)} \\ |v_i(t)| &= \sqrt{\frac{A}{R}\beta\left(\frac{A}{\alpha_0 R^2}\right)} \end{aligned} \tag{3.21}$$

3.3 Modified Newtonian Dynamics Via Gravity Potential

Another way to modified Newtonian dynamics equation is trough gravity potential. This motivation has solved in some paper [11]. This motivation usually called as AQUAL or the other name as QUMOND [12]. We will show the stepping by writing Newtonian dynamics equation,

$$\begin{aligned} F &= ma \\ m_i G\sum_{\substack{j=1\\ j\neq i}}^{N}\frac{M\left(x_j - x_i\right)}{|x_j - x_i|^3} &= m_i\ddot{x} \\ m_i f_i &= m_i\ddot{x} \\ \ddot{x} &= f_i \end{aligned} \tag{3.22}$$

Of course $f \equiv G\sum_{\substack{j=1\\ j\neq i}}\frac{M(x_j-x_i)}{|x_j-x_i|^3}$. Then another way to get this formulation by fundamental gravity potential,

$$E(x) := \frac{1}{4\pi}\frac{1}{|x|} \ (x \in \mathbb{R}^3, x \neq 0). \tag{3.23}$$

We have a solution by Poisson's equations in the case of acceleration due to an attracting massive object. We write the property as

$$-\Delta E = \delta \ (\text{Dirac's } \delta) \tag{3.24}$$

where the notation $\Delta = \left(\frac{\partial}{\partial X^{(1)}}\right)^2 + \left(\frac{\partial}{\partial X^{(2)}}\right)^2 + \left(\frac{\partial}{\partial X^{(3)}}\right)^2$ or sometime use Laplacian operator ∇^2. Again we define $u(x) := E * f(x) := \int_{\mathbb{R}^3} E(x-y)f(y)dy$. This notation called as convolution of E and f.

$$\begin{cases} -\Delta u = f \\ u(x) = \mathcal{O}\left(\frac{1}{|x|}\right) \text{at} x \to \infty \end{cases} \tag{3.25}$$

So we can write the convolution calculation as follows

$$\begin{aligned} u(x) = E * \delta_{x_0}(x) &= \text{"} \int_{\mathbb{R}^3} E(x-y)\delta_{x_0}(y)dy\text{"} \\ &= E(x-x_0) = \tfrac{1}{4\pi}\tfrac{1}{|x-x_0|}. \end{aligned} \tag{3.26}$$

Next we consider for N-particles case $\{x_i\}_{i=1}^N \subset \mathbb{R}^3$, so the force we have

$$\begin{cases} f &= \sum_{i=1}^N m_i \delta_{x_i}, -\delta U = f \\ u(x) &= E * \left(\sum_{i=1}^N m_i \delta_{x_i}\right) \\ &= \sum_{i=1}^N m_i E * \delta_{x_i} = \sum_{i=1}^N m_i E(x-x_i) = \sum_{i=1}^N \frac{m_i}{4\pi}\frac{1}{|x-x_i|} \end{cases} \tag{3.27}$$

Therefore for gravity force at $x \neq x_i$

$$\begin{aligned} \nabla\left(4\pi G u(x)\right) &= 4\pi G \nabla U(x) \\ &= 4\pi G \sum_{j=1}^{N} \frac{m_j}{4\pi} \nabla E\left(x - x_j\right) \\ &= 4\pi \sum_{j=1}^{N} m_j G \nabla E\left(x - x_j\right). \end{aligned} \tag{3.28}$$

where we have gravity potential

$$\begin{aligned} \nabla E(x) &= -\tfrac{1}{4\pi} \tfrac{x}{|x|^3}. \\ &= -G \textstyle\sum_{j=1}^{N} m_j \frac{x - x_j}{|x - x_j|^3} \\ &= G \textstyle\sum_{j=1, j\neq i}^{N} m_j \frac{x_j - x}{|x_j - x|^3}. \end{aligned} \tag{3.29}$$

at $x = x_i$; $\sum_{\substack{j=1 \\ j\neq i}}^{N}$ for

$$\mathrm{ND} \approx \begin{cases} \ddot{x}_i & = 4\pi G \nabla u_N(x_i) \\ -\delta u_N & = \sum_{\substack{j=1 \\ (j\neq 1)}}^{N} m_j \delta_{x_j} \\ u(x) & = \mathcal{O}\left(\frac{1}{|x|}\right) \quad \text{as } |x| \to \infty \end{cases} \tag{3.30}$$

Chapter 4

Formal Equation of Newtonian Dynamics

4.1 Introduction

Basically in order to explain the particle motions we can consider to look at the case as 'group' of particles that move each times. The 'group' can look as field that indicated the distribution of particles. Start from this condition, we can see that the basic law that work there such a continuity motion case. Therefore we look at a new analog where the particle motion is essentially as a possible solution in our simulation.

Actually continuity equation governs the conservation of mass/charge/ probability of any closed system. This equation provides us with information about the system. The information is carried from one point to another by a particle (field) wave. In this case the close system is initially given by inside area of radius galaxy. The motivation to work in this way is actually because the dynamical equation of Newton's law, can be obtain by gradient of potential. Therefore another Newtonian modified can also be obtaied by this way. Then, this section we will show another way to modified Newtonian's dynamics such as MOND in another expression called AQUAL.

4.2 Analogize by Field of Particles Motions

In this section we will show the formalism of Newtonian dynamics by using another way. This is the formulation of formal many particles limit for $\mathrm{N}_{particles}\{x_i^N(t)\}_{i=1}^N \subset \mathbb{R}^d$

$$\begin{aligned} m_i^{(N)} &= m_i > 0; \qquad \text{mass of } x_i^N(t) \qquad (\text{depend on } N) \\ \sum_{i=1}^{N} m_i &= M \qquad (\text{Independent of } N) \end{aligned} \tag{4.1}$$

We suppose for these conditions

1. $\underline{\rho(x,t) \geq 0 \text{ limit density as N} \to \infty}$
   for any $\varphi \in C\left(\mathbb{R}^d\right)$,
   $\lim_{N\to\infty}\left(\sum_{i=1}^N m_i\varphi\left(x_i^N(t)\right)\right) = \int_{\mathbb{R}^d}\varphi(x)\rho(x,t)dx.$
2. $\underline{\upsilon(x,t) \in \mathbb{R}^d; \text{ limit velocity field.}}$
   $^{\exists}\upsilon^N(x,t) \in \mathbb{R}^d$ such that $\dot{x}_i^N(t) = \upsilon^N\left(x_i(t),t\right)$,
   and $\lim_{N\to\infty}\upsilon_N(x,t) = \upsilon(x,t)$.
3. $\underline{a(x,t) \in \mathbb{R}^d; \text{ limit acceleration field.}}$
   $a(x,t) \in \mathbb{R}^d$ such that $\ddot{x}_i(t) = a^N(x_i(t),t)$
   and $\lim_{N\to\infty} a^N(x,t) = a(x,t)$.

Since we consider the condition

$$\begin{aligned} C_0^\infty(\mathbb{R}^d) &= \{\varphi : \mathbb{R}^d \to \mathbb{R}|_{bounded\ support}^{C^\infty - class}\} \\ &= \{\varphi \in C^\infty(\mathbb{R}^d)|^{\exists} R > 0, such that \varphi(x) \to (|X| \geq R)\} \end{aligned} \tag{4.2}$$

Then we will give a test function for these term

1. For any $\varphi \in C_0^\infty(\mathbb{R}^d)$. Then, we denote as

$$\begin{aligned}\frac{d}{dt}\{\sum_{i=1}^{N} m_i\varphi(x_i^N(t))\} &= \textstyle\sum_{i=1}^{N} m_i\nabla\varphi(x_i^N(t))\cdot\dot{x}_i^N(t)\\ &= \textstyle\sum_{i=1}^{N} m_i\nabla\varphi(x_i^N(t))v^N(x_i^N,t). \qquad (4.3)\end{aligned}$$

Note 1 : we consider $x = \begin{pmatrix} x^{(1)} \\ x^{(2)} \\ x^{(3)} \end{pmatrix}$, $d = 3$ and $\varphi \in C_0^\infty(\mathbb{R}^d)$.

$$\begin{aligned}\frac{d}{dt}\varphi\left(x(t)\right) &= \sum_{j=1}^{d}\frac{\partial\varphi}{\partial x^{(i)}}\left(x(t)\right)\dot{x}^{(j)}(t)\\ &= \nabla\varphi\left(x(t)\right)\cdot\dot{x}(t). \qquad (4.4)\end{aligned}$$

Then we continue the calculation of 4.3 by taking $N \to \infty$ as formally case. So we have

$$\frac{d}{dt}\int_{\mathbb{R}^d}\varphi(x)\rho(x,t)dx = \int_{\mathbb{R}^d}\nabla\varphi(x)\{v(x,t)\rho(x,t)\}dx. \qquad (4.5)$$

Note 2 : Integration by part for $\varphi \in C_0^1(\mathbb{R}^d), \psi \in C^1\left(\mathbb{R}^d, \mathbb{R}^d\right)$

$$\int_{\mathbb{R}^d}\nabla\varphi(x)\cdot\psi(x)dx = -\int_{\mathbb{R}^d}\varphi(x)\ \mathrm{div}\ \psi(x)dx. \qquad (4.6)$$

where $\left(\mathrm{div}\ \psi = \sum_{i=1}^{d}\frac{\partial\psi_j}{\partial x^{(j)}} \in \mathbb{R}^d\right)$.
Example 1D case : $\int_a^b \varphi'(x)\psi(x)dx = [\varphi(x)\psi(x)]_a^b - \int_a^b \varphi(x)\psi'(x)dx$.
Meanwhile in our case eq.(4.6) we only have the $2nd$ term. It is caused we only consider the integral value of inside the radius, where $1st$ is indicated the outside area. So $1st$ term is equal to 0.

Therefore the integral of eq.(4.4) can show as

$$(l.h.s) = \int_{\mathbb{R}^d} \varphi(x)\rho(x,t).$$
$$(r.h.s) = -\int_{\mathbb{R}^d} \varphi(x)\text{div } \left(\rho(x,t)\upsilon(x,t)\right) dx.$$

We forward the right side into left side, then we have

$$\int_{\mathbb{R}^d} \varphi(x)\{\frac{\partial \rho}{\partial t}(x,t) + \text{div } \left(\rho(x,t)\upsilon(x,t)\right)\}dx = 0 \quad \left(\forall \varphi \in C_0^\infty(\mathbb{R}^d)\right). \tag{4.7}$$

Since $\varphi(x) \neq 0$, so $\{\} = 0$. Then we have relation

$$\frac{\partial \rho}{\partial t} + \text{div } (\rho\upsilon) = 0. \tag{4.8}$$

As we know the last expression is familiar equation that we commonly known as continuity equation or related by mass conservation.

2. For any $\psi \in C_0^\infty(\mathbb{R}^d, \mathbb{R}^d)$. Then we denote as,

$$\begin{aligned}\frac{d}{dt}\left\{\sum_{i=1}^N m_i \psi(x_i^N(t)) \cdot \dot{x}_i^N(t)\right\} &= \sum_{i=1}^N m_i \left\{\left((\dot{x}_i^N(t)\cdot\nabla)\psi\right)(x_i^N(t)) \cdot \dot{x}_i^N(t) + \right. \\ &\left. \psi\left(x_i^N(t)\right)\cdot \dot{x}_i^N(t)\right\}. \\ &= \sum_{i=1}^N m_i \left\{\left(\left(\upsilon_i^N(x_i^N(t),t)\cdot\nabla\right)\psi\right)(x_i^N(t)) \cdot \upsilon^N\left(x_i^N(t),t\right) + \right. \\ &\left. \psi\left(x_i^N(t)\right)\cdot a^N\left(x_i^N(t),t\right)\right\}. \end{aligned} \tag{4.9}$$

Then we taking at $N \to \infty$,

$$\begin{aligned}\frac{d}{dt}\left\{\int_{\mathbb{R}^d} \psi(x)\cdot\upsilon(x,t)\rho(x,t)dx\right\} &= \int \{((\upsilon(x,t)\cdot\nabla)\psi(x))\,\upsilon(x,t) \\ &+\psi(x)\cdot a(x,t)\}\,\rho(x,t)dx. \end{aligned} \tag{4.10}$$

Than same as the previous calculation, we integrate it by part and we get

$$\begin{aligned}
\int_{\mathbb{R}^d} \psi \frac{\partial}{\partial t}(\rho \upsilon) dx &= -\int_{\mathbb{R}^d} \psi \cdot \{\text{div } (\rho\upsilon)\upsilon + \rho(\upsilon \cdot \nabla)\upsilon\}\, dx. \\
&\quad + \int_{\mathbb{R}^d} \psi \cdot (\rho a) dx \qquad\qquad ({}^{\forall}\psi \in C_0^{\infty}). \\
\therefore \frac{\partial}{\partial t}(\rho\upsilon) &= -\text{div } (\rho\upsilon)\upsilon - \rho(\upsilon \cdot \nabla)^{\upsilon} + \rho a. \\
\cancel{-(\text{div}\rho\upsilon)\upsilon} + \cancel{\rho}\frac{d\upsilon}{dt} &= \cancel{-\text{div } (\rho\upsilon)\upsilon} - \cancel{\rho}(\upsilon \cdot \nabla)^{\upsilon} + \cancel{\rho} a.
\end{aligned}$$

So, we get the result

$$\frac{\partial \upsilon}{\partial t} \quad = a - (\upsilon \cdot \nabla)\upsilon \frac{\partial \upsilon}{\partial t} + (\upsilon \cdot \nabla)\upsilon \quad =: D_t \upsilon. \quad . \tag{4.11}$$

Where $D_t \upsilon = a$. The last equation 4.11, is commonly called as *material derivative*. Furthermore we can give the numerical work by this equation like this

$$\begin{aligned}
D_t \upsilon(x(t), t) &= \frac{d}{dt}\upsilon(x(t), t) \\
\text{If } \dot{x}(t) &= \upsilon(x(t), t)
\end{aligned} \tag{4.12}$$

So, generally we can write like this notation

$$\begin{cases}
D_t = a. \\
\rho_t + \text{div } (\rho\upsilon) = 0.
\end{cases} \tag{4.13}$$

Then, the implementation for each theory that we propose in this work, such as Newtonian dynamics, MOND dynamics, and AQUAL, will be shown like this

$$\text{ND}: \quad \ddot{x}_i(t) = G\sum_{\substack{j=1\\ j\neq i}}^{N} \boxed{\frac{m_j\,(x_j(t)-x_i(t))}{|x_j(t)-x_i(t)|^3}} =: f_i(t).$$

$$a(x,t) = \lim_{N\to\infty} a^N(x,t). \tag{4.14}$$

$$= \int_{\mathbb{R}^d} G\frac{y-x}{|y-x|^3}\rho(y)dy. \qquad for\ \ y = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \end{pmatrix} \tag{4.15}$$

So, we can simplify the notation for ND case as follows

$$\begin{cases} D_t\upsilon(x,t) = a(x,t) \\ a(\cdot,t) = 4\pi G\ \ E * \rho(\cdot,t) \\ \rho_t(x,t) + \text{div}\ \ (\rho(x,t),\upsilon(x,t)) \\ \rho(x,0) = \rho_0(x) \geqq 0 \\ \upsilon(x,0) = \upsilon_0(x) \in \mathbb{R}^d. \end{cases} \tag{4.16}$$

Then, for MOND case and AQUAL case will be given sequentially as follows

$$\text{MOND}: \quad \mu\left(\frac{|\ddot{x}_i|}{\alpha_0}\right)\ddot{x}_i = f_i \Longleftrightarrow \ddot{x}_i = \beta(f_i) = a^N(x_i^N,t).$$

$$a(x,t) = \beta(4\pi G\nabla(E*\rho)). \tag{4.17}$$

$$\text{AQUAL}: \qquad a = \nabla U$$

$$\text{-div}\ \ \left(\mu\left(\frac{|\nabla U|}{\alpha_0}\right)\nabla U\right) = 4\pi G\rho. \tag{4.18}$$

Of course for $\beta \equiv \frac{|f_i(X(t))|}{\alpha_0}$. These our modified from Newtonian dynamics equation.

Chapter 5

Numerical Scheme

5.1 Briefly Numerical Analysis and It's Mathematical Notations

Newtonian dynamic and its modified, basically is problem in ordinary differential case. Sometimes we can calculate by manually. But in order to get much calculation, we need work hard very complicated to solve it. So we will solve this problem by numerical calculation.

So many kind of numerical method to solve ordinary differential equation (ODE), but for easiest way we can use by Euler method. This method maybe is ot really an efficient method, but many of the ideas involved in the numerical solution of differential equations are introduced most simply with it.

Before we discuss more about the numerical solution in this work. It's important to know the mathematical notation that we are going to use it in simulation. So, this part we will give an mathematical overview for this system. Let's start by giving notation for the particles. We write the particle as

$$X = \{x_i\}_{i=1}^{\mathrm{N}} \quad \subset \mathbb{R}^{\mathrm{d}\times\mathrm{N}} \tag{5.1}$$

where X is particle for i as index of each particles as much as N. Then we set

for dynamically system, $f_i(X) := \frac{F_i(X)}{m_i}$, and for indicating the step we show

$$\text{ND}\begin{cases} \ddot{x}(t) = f_i(X(t)) & (0 \le t \le T), \\ \dot{x}(t) = v_i(t) & (0 \le t \le T), \\ x_i(0) = x_i^0 \ \text{; given} & (i = 1, \dots \text{N}) \\ v_i(0) = v_i^0 \ \text{; given} & (i = 1, \dots \text{N}) \end{cases} \tag{5.2}$$

By these notation, we will use it into the program.

5.2 Discretization and The Numerical Scheme

After we fix it the mathematical notation, we also need to attend the time stepping. It is mean that our simulation is depending the function of time, so should fix it. We choose $\Delta t > 0, K := \left[\frac{T}{\Delta t}\right]$, and we consider for our case

$$x_i^k = \begin{pmatrix} x_{\text{i1}}^{\text{k}} \\ x_{\text{i2}}^{\text{k}} \end{pmatrix} \sim x_i(k\Delta t) \tag{5.3}$$

$$v_i = \begin{pmatrix} v_{\text{i1}}^{\text{k}} \\ v_{\text{i2}}^{\text{k}} \end{pmatrix} \sim v_i(k\Delta t) \tag{5.4}$$

where we give notation $(k = 0, 1, \dots, \text{K}, i = 1, \dots, \text{N})$.
Time stepping k start from initially, which mean 0, and will be end for K time. It will be implemented into i-th particle. In our code programe we have this conditions

$$\text{X}^{\text{old}} = (x_i^{\text{old}})_{\text{i=1}}^{\text{N}} \text{; V}^{\text{old}} = (v_i^{\text{old}})_{\text{i=1}}^{\text{N}} \tag{5.5}$$

$$\text{X}^{\text{new}} = (x_i^{\text{new}})_{\text{i=1}}^{\text{N}} \text{; V}^{\text{new}} = (v_i^{\text{new}})_{\text{i=1}}^{\text{N}} . \tag{5.6}$$

So this mean that we don't store the previous data. We overwrite the data by new update data of position x and velocity v.

5.2.1 Euler Scheme

As we said before, the numerical method in order to calculate the differential problem is used Euler method. Therefore we should know how actually this method can work. So we will show the numerical scheme for Euler method in this notation

$$\begin{cases} \frac{x_i^{k+1}-x_i^k}{\Delta t} = v_i^k \\ \frac{v_i^{k+1}-v_i^k}{\Delta t} = f_i(x^k) \\ x_i^0 \quad ; v_i^0 \quad ; \text{given.} \end{cases} \tag{5.7}$$

This scheme will be implemented into the program and we discussing more about the result will be comparing by observation data.

5.2.2 Symplectic scheme

Another scheme that we used in this simulation is symplectic scheme. Generally this method it's better than Euler method. The main reason is about the symplectic always give new velocity update for each step position. So, the result might be closely than the exact.

Then, this is the symplectic scheme which used in the program

$$\begin{cases} \frac{v_i^{k+1}-v_i^k}{\Delta t} = f_i(x^k) \\ \frac{x_i^{k+1}-x_i^k}{\Delta t} = v_i^{(k+1)} \\ x_i^0 \quad ; v_i^0 \quad ; \text{given.} \end{cases} \tag{5.8}$$

Based two numerical method we will show how more accurate between Euler and symplectic method.

5.2.3 Initialization of particles

After we give an uniform scale for our case, the next step is make a condition that we consider as initial position. By this table will be shown some of the initialization By using the table initialization above we can do the simulation

Initialization	Physical meaning	Unity	Value
m	Mass of particle	1	$M_\odot$
R	Radius	2.9	Kpc
ω	angular velocity	Random value$(0, \pi)$	1 galyear
$x_i^j\,(i,j)^\dagger$	position of particles	Random uniform	Kpc
$v_i^j\,(i,j)$	velocity	$(\text{-}v\omega\sin(\omega t), v\omega\cos(\omega t))$	m/s^2
t_0	time initialization	0	$gyear$

for this system.

5.3 Galaxy scaling for Simulation

Galaxy as an object which work in large scale, certainly need big memory to store it. This section we want to show the new scaling in our simulation, that we consider to convert in small scale. Firstly we convert the time scale. In the real observation galaxy time scale in order to indicate one period the galaxy rotations is called galaxy year -sometimes cosmic year. This mean one galaxy year is about $1 \times 10^{15}s$. Basically if use the real scaling of galaxy, it will be crowded enough to calculate it. So we need to convert this scale into small dimension scale as given this calculation

$$\tilde{t} = \frac{t}{C_1} \tag{5.9}$$

for $\tilde{t}$ is indicated by time after re-scaling and we bring out $C_1 = 10^{15}$ as value of galaxy year. So, we can keep C_1 value from calculation and sometimes if we need to show, we can use it in the data that we obtained.

Then for uniformly the scale of galaxy object, we write into converting table this

Table 5.1: Table of conversion

Variable & constant	Physical Meaning	(m, Kg, s) scale	$(Kpc, M_{\odot}, gy)$ scale
G	Gravity constant	6.67×10^{-11}[Kg.m^{-1}.s^{-2}]	13.37×10^{-11}[Kpc.M$_{\odot}^{-1}$.galyear^{-2}
M_{total}	Mass of the total particles	1 ($\frac{10^{11}}{N}$) [Kg]	1 [$M_{\odot}$]
t	Time scale	1 [$\times 10^{15}$]	1 [gyear]
R	Radius of Milky way	2.9($\times 10^{20}$ [m]	5[Kpc]
v	Velocity	1.3×10^{3}	4.2$\times 10^{-17}$ [Kpc]

Finally we get the uniform dimension like on the table(5.1) that we can do for further step with give an initialization of this simulation.

Chapter 6

Numerical Result & Discussion

In this work we consider for $N = 10$ particles. The first simulation we give $\Delta t = 0.001$ for K as a total of iteration is $K = 5000$. In this case we construct by $K = \frac{t}{\Delta t}$ where $T = 5$. For this ω we set as a radius velocity of the maximum position in random number between $(0, \pi)$.

By using the initialization that we give previous section in the table.(5.1), we do the simulations by two methods. We consider to use Euler and symplectic method as a numerical solver this case ,

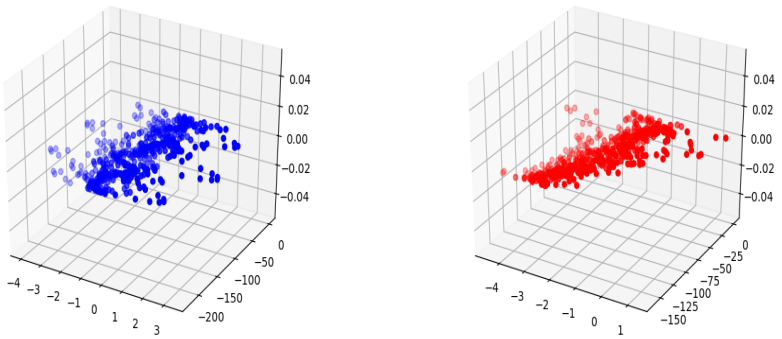

Figure 6.1: Graph from Euler (blue) & symplectic method (red)

Figure (6.1) obtained as Euler method for blue scatter and symplectic for red scatter. By those figures we can say that our program has success work. But

It's should be analyze more about the gravity and dynamically of particle. We can also check by comparing with observation that will be more precisely to conclude this simulation.

Basically, currently we still only show for Newtonian dynamics simulation. We should necessary fix for MOND simulation, beside we show the comparing with observation data result. Then, afterward we can really conclude which one the best method to do simulation if we compare with all of considerations.

Appendix A

Program codes

1. Plot Two Particles

```
import matplotlib.pyplot as plt
import numpy as np
from time import time
from numpy import sqrt

start = time()
def phi_inv(q):
    return sqrt((q**2+sqrt(1+q**2))/2.0)**(-1)
def phi_inverse(q):
    return sqrt(q)*sqrt((1+sqrt(1+(4/q**2)))/2.0)
#Global parameter
a_0 = 10e-11
m = 10e11 #[MO]
R = np.arange(0.05,50,0.001) #[kpc]
G = 13.34*10e-11 #[kpc^3 MO^-1 gy^-2]
#Newton Dynamic
omega = sqrt((G*m)/(4*R**3)) #velocities
vt = omega*R
plt.xlabel("R [kpc]")
plt.ylabel("vt")
plt.plot(R,omega, 'r-', label="Newtonian Dynamics")
#MoND
omega = sqrt(G*m)*R**(-(7/2)) #velocities
omega1 = sqrt((a_0/R)*sqrt(m*G/(a_0*4*R**3)))
vt = omega*R
vt1 = omega1*R
#plt.plot(R,vt, 'b-', label="Modified Newtonian Dynamics")
plt.plot(R,omega1, 'b-', label="Modified Newtonian Dynamics")
#plt.title("Total Energy of Symplectic Newton Dynamics")
#filename='./figures/plot.png'
```

```
31 #plt.savefig(filename)
32 print("Time for running ", time()-start, "seconds")
33 plt.legend()
34 plt.show()
```

2. Two Particles Cases (ND)

```
 1 import matplotlib.pyplot as plt
 2 from numpy import sin,cos,pi,sqrt,exp,floor,zeros,copy,array
 3 from numpy.random import normal
 4 from numpy.linalg import norm
 5 from random import uniform
 6 from time import time
 7
 8 start = time()
 9 def euler(x,v):
10     for i in range(n_particles):
11         sigmaF = zeros(2)
12         for j in range(n_particles):
13                 if(i!=j):
14                     sigmaF += f(x[i],x[j])
15         x[i] += v[i]*dt
16         v[i] += sigmaF*dt
17
18 def euler_edit(x,v):
19     for i in range(n_particles):
20         sigmaF = zeros(2)
21         x[i] += v[i]*dt
22         for j in range(n_particles):
23                 if(i!=j):
24                     sigmaF += f(x[i],x[j])
25         v[i] += sigmaF*dt
26
27 def symplectic(x,v):
28     for i in range(n_particles):
29         sigmaF = zeros(2)
30         for j in range(n_particles):
31                 if(i!=j):
32                     sigmaF += f(x[i],x[j])
33         v[i] += sigmaF*dt
34         x[i] += v[i]*dt
35
36 def symplectic_edit(x,v):
37     for i in range(n_particles):
38         sigmaF = zeros(2)
39         x[i] += v[i]*dt
40         for j in range(n_particles):
```

```
41                  if(i!=j):
42                      sigmaF += f(x[i],x[j])
43          v[i] += sigmaF*dt
44          x[i] += v[i]*dt
45
46 def f(xi,xj):
47      rij = xj-xi
48      return (G*m*rij)/(norm(rij)+epsilon)**3
49 def init_two():
50      x1 = ([R*cos(omega*0),R*sin(omega*0)])
51      x2 = -copy(x1)
52      v1 = ([-omega*x1[1],omega*x1[0]])
53      v2 = -copy(v1)
54      x = array([x1,x2])
55      v = array([v1,v2])
56      return x,v
57 def kinetic_energy():
58      ke = 0.0
59      for i in range(n_particles):
60          ke += 0.5*m*norm(v[i])**2
61      return ke
62 def potential_energy():
63      pe = 0.0
64      i,j = 0,0
65      while(1<=i<j<=n_particles):
66          rij = x[j]-x[i]
67          pe += (G*m*m)/(norm(rij))
68          i += 1
69          j += 1
70      return pe
71 #Global parameter
72 n_particles = 2 #particles
73 d = 2 #dimension
74 m = 10e11/n_particles #[MO]
75 R = 2.9 #[kpc]
76 G = 13.34*10e-11 #[kpc^3 MO^-1 gy^-2]
77 omega = sqrt((G*m)/(4*R**3)) #velocities
78 #omega = normal(0,2*pi) #velocities
79 epsilon = 1e-3
80 T = 100
81 dt = 0.001
82 N = int(floor(T/dt))
83 scale = 30.0
84 #initial condition
85 x,v = init_two()
86 #x = get_init_coordinates()
87 #v = get_init_velocities()
```

```
print(x)
#main loop
#plt.plot(x[:,0],x[:,1], 'ro')
en_total = zeros(N)
k_array = zeros(N)
for k in range(N):
    euler(x,v)
    #en_total[k] = kinetic_energy()-potential_energy()
    #k_array[k] = k*dt
    #plt.plot(xe[:,0],xe[:,1], 'b.')
    #plt.xlim(right=scale,left=-scale)
    #plt.ylim(top=scale,bottom=-scale)
    #plt.axes(aspect='equal')
    if(k%100==0):
        plt.plot(x[:,0],x[:,1], 'b.')
#plt.xlabel("T [kpc]")
#plt.ylabel("Total Energy")
#plt.title("Total Energy of Symplectic Newton Dynamics")
#plt.plot(k_array,en_total, 'g-')
#filename='./figures/plot.png'
#plt.savefig(filename)
print("Time for running ", N, "iteration :", time()-start, "seconds")
print(x)
plt.show()
```

3. Two Particles Cases (MOND)

```
import matplotlib.pyplot as plt
from numpy import sin,cos,pi,sqrt,exp,floor,zeros,copy,array
from numpy.random import normal
from numpy.linalg import norm,inv
from random import uniform
from time import time

start = time()
def euler(x,v):
    for i in range(n_particles):
        sigmaF = zeros(2)
        for j in range(n_particles):
                if(i!=j):
                    sigmaF += f(x[i],x[j])
        x[i] += v[i]*dt
        v[i] += (a_0*phi_inv(norm(sigmaF)/(a_0))*(sigmaF/norm(sigmaF)))*
    dt

def euler_edit(x,v):
    for i in range(n_particles):
```

```
20          sigmaF = zeros(2)
21          x[i] += v[i]*dt
22          for j in range(n_particles):
23                  if(i!=j):
24                      sigmaF += f(x[i],x[j])
25          v[i] += (a_0*(phi_inv(norm(sigmaF)/(m*a_0))*(sigmaF/norm(sigmaF)
        )))*dt
26
27 def symplectic(x,v):
28      for i in range(n_particles):
29          sigmaF = zeros(2)
30          for j in range(n_particles):
31                  if(i!=j):
32                      sigmaF += f(x[i],x[j])
33          v[i] += G*sigmaF*dt
34          x[i] += v[i]*dt
35 def f(xi,xj):
36      rij = xj-xi
37      return (G*m*rij)/(norm(rij)+epsilon)**3
38 def init_two():
39      x1 = ([R*cos(omega*0),R*sin(omega*0)])
40      x2 = -copy(x1)
41      v1 = ([-omega*x1[1],omega*x1[0]])
42      v2 = -copy(v1)
43      x = array([x1,x2])
44      v = array([v1,v2])
45      return x,v
46 def kinetic_energy():
47      sigmaN = 0.0
48      for i in range(n_particles):
49          sigmaN += 0.5*m*norm(v[i])**2
50      return sigmaN
51 def phi_inv(q):
52      return sqrt((q**2+sqrt(1+q**2))/2.0)**(-1)
53 #Global parameter
54 n_particles = 2 #particles
55 d = 2 #dimension
56 m = 10e11/n_particles #[MO]
57 R = 2.9 #[kpc]
58 G = 13.34*10e-11 #[kpc^3 MO^-1 gy^-2]
59 omega = sqrt((G*m)/(4*R**3)) #velocities
60 epsilon = 1e-3
61 T = 100
62 dt = 0.001
63 N = int(floor(T/dt))
64 scale = 30.0
65 a_0 = 10e-2
```

```
66 #initial condition
67 x,v = init_two()
68 #x = get_init_coordinates()
69 #v = get_init_velocities()
70 print(x)
71 #main loop
72 #plt.plot(x[:,0],x[:,1], 'ro')
73 for k in range(N):
74     euler(x,v)
75     #print(kinetic_energy())
76     #plt.plot(xe[:,0],xe[:,1], 'b.')
77     #plt.xlim(right=scale,left=-scale)
78     #plt.ylim(top=scale,bottom=-scale)
79     #plt.axes(aspect='equal')
80     if(k%100==0):
81         plt.plot(x[:,0],x[:,1], 'b.')
82 #filename='./figures/plot.png'
83 #plt.savefig(filename)
84 print("Time for running ", N, "iteration :", time()-start, "seconds")
85 print(x)
86 plt.show()
```

4. Many Particles (ND)

```
1 import matplotlib.pyplot as plt
2 from numpy import sin,cos,pi,sqrt,exp,floor,zeros,copy,array
3 from numpy.random import normal
4 from numpy.linalg import norm
5 from random import uniform
6 from time import time
7
8 start = time()
9 def euler(x,v):
10     for i in range(n_particles):
11         sigmaF = zeros(2)
12         for j in range(n_particles):
13                 if(i!=j):
14                     sigmaF += f(x[i],x[j])
15         x[i] += v[i]*dt
16         v[i] += G*sigmaF*dt
17 def symplectic(x,v):
18     for i in range(n_particles):
19         sigmaF = zeros(2)
20         for j in range(n_particles):
21                 if(i!=j):
22                     sigmaF += f(x[i],x[j])
23         v[i] += G*sigmaF*dt
```

```
24          x[i] += v[i]*dt
25 def get_init_coordinates():
26      x = zeros((n_particles,d))
27      i = 0
28      while(i<n_particles):
29          x1 = normal(-R,R)
30          x2 = normal(-R,R)
31          if(abs(x1**2+x2**2)<R**2):
32              x[i] = ([x1,x2])
33              i+=1
34          else:
35              i=i
36      return x
37 def get_init_velocities():
38      v = zeros((n_particles,d))
39      for i in range(n_particles):
40          v[i] = ([omega*x[i,1],omega*x[i,0]])
41      return v
42 def f(xi,xj):
43      rij = xj-xi
44      return (m*(rij))/(norm(rij)+epsilon)**3
45 def kinetic_energy():
46      sigmaN = 0.0
47      for i in range(n_particles):
48          sigmaN += 0.5*m*norm(v[i])**2
49      return sigmaN
50 #Global parameter
51 n_particles = 10 #particles
52 d = 2 #dimension
53 m = 10e11/n_particles #[MO]
54 R = 2.9 #[kpc]
55 G = 13.34*10e-11 #[kpc^3 MO^-1 gy^-2]
56 omega = sqrt((G*m)/(4*R**3)) #velocities
57 epsilon = 1e-3
58 T = 5
59 dt = 0.001
60 N = int(floor(T/dt))
61 scale = 30.0
62 #initial condition
63 x = get_init_coordinates()
64 v = get_init_velocities()
65 print(x)
66 #main loop
67 plt.plot(x[:,0],x[:,1], 'ro')
68 for k in range(N):
69      euler(x,v)
70      #print(kinetic_energy())
```

```
71      #plt.plot(xe[:,0],xe[:,1], 'b.')
72      #plt.xlim(right=scale,left=-scale)
73      #plt.ylim(top=scale,bottom=-scale)
74      #plt.axes(aspect='equal')
75      if(k%100==0):
76          plt.plot(x[:,0],x[:,1], 'b.')
77 #filename='./figures/plot.png'
78 #plt.savefig(filename)
79 print("Time for running ", N, "iteration :", time()-start, "seconds")
80 print(x)
81 plt.show()
```

5. Many Particles (MOND)

```
1  import matplotlib.pyplot as plt
2  from numpy import sin,cos,pi,sqrt,exp,floor,zeros
3  from numpy.random import normal
4  from numpy.linalg import norm
5  from random import uniform
6  from time import time
7
8  start = time()
9  def euler_bound(x,v):
10     for i in range(n_particles):
11         x[i] += v[i]*dt
12         for j in range(n_particles):
13             if(i!=j):
14                 a = norm(f(x[i],x[j]))
15                 muv = mu(a/a_0)
16                 #print(a,muv)
17                 v[i] += (f(x[i],x[j])/muv)*dt
18         if (norm(x[i]) > R) or (norm(x[i]) < -R):
19             v[i] = -v[i]
20             x[i] += v[i]*dt
21 def euler(x,v):
22     x_k = x
23     for i in range(n_particles):
24         x[i] += v[i]*dt
25         for j in range(n_particles):
26             if(i!=j):
27                 a = norm(f(x_k[i],x_k[j]))
28                 muv = mu(a/a_0)
29                 #print(a,muv)
30                 v[i] += (f(x_k[i],x_k[j])/muv)*dt
31 def euler_edit(x,v):
32     for i in range(n_particles):
33         sigmaF = zeros(2)
```

```
34          x[i] += v[i]*dt
35          for j in range(n_particles):
36                  if(i!=j):
37                      sigmaF += f(x[i],x[j])
38          v[i] += (a_0*phi_inv(norm(sigmaF)/(m*a_0))*(sigmaF/norm(sigmaF))
        )*dt
39
40
41  def symplectic(x,v):
42      for i in range(n_particles):
43          x[i] += v[i]*dt
44          for j in range(n_particles):
45              if(i!=j):
46                  v[i] += f(x[i],x[j])*dt
47  def get_init_coordinates():
48      x = zeros((n_particles,d))
49      i = 0
50      while(i<n_particles):
51          x1 = normal(-R,R)
52          x2 = normal(-R,R)
53          if(abs(x1**2+x2**2)<R**2):
54              x[i] = ([x1,x2])
55              i+=1
56          else:
57              i=i
58      return x
59  def get_distribution():
60      x = get_init_coordinates()
61      rho = zeros((n_particles,d))
62      i = 0
63      while(i<n_particles):
64          #rho[i] = [(M_total/(pi*R**2))*exp(x[i,0]/R)*sech((x[i,1]/R))
        **2,(M_total/(pi*R**2))*exp(x[i,0]/R)*sech((x[i,1]/R))**2]
65          i += 1
66      return rho,x
67
68  def get_init_velocities():
69      v = zeros((n_particles,d))
70      for i in range(n_particles):
71          v[i] = ([-(1/sqrt(rho[i,0]))*omega*sin(omega*t),(1/sqrt(rho[i
        ,1]))*omega*cos(omega*t)])
72      return v
73  def f(xi,xj):
74      rij = xj-xi
75      return (G*m*(rij))/(norm(rij)+epsilon)**3
76  def mu(s):
77      return s/sqrt(1+s**2)
```

```
78 def phi_inv(q):
79     return sqrt((q+sqrt(1+q**2))/2.0)**(-1)
80
81 #Global parameter
82 n_particles = 10 #particles
83 d = 2 #dimension
84 m = 10e11/n_particles #[MO]
85 M_total = m*n_particles
86 R = 2.9 #[kpc]
87 G = 13.34*10e-11 #[kpc^3 MO^-1 gy^-2]
88 omega = normal(0,2*pi) #velocities
89 epsilon = 1e-8
90 T = 5
91 dt = 0.01
92 t = 0. #step
93 N = int(floor(T/dt))
94 scale = 7.0
95 a_0 =  1e-10
96 #initial condition
97 #x,v = init_two()
98 rho,x = get_distribution()
99 plt.plot(x[:,0],x[:,1], 'r.')
100 v = get_init_velocities()
101 #main loop
102 for k in range(N):
103     t = k*dt
104     euler_edit(x,v)
105     #plt.plot(xe[:,0],xe[:,1], 'b.')
106     plt.xlim(right=scale,left=-scale)
107     plt.ylim(top=scale,bottom=-scale)
108     #plt.axes(aspect='equal')
109     plt.plot(x[:,0],x[:,1], 'b.')
110 #filename='./figures/plot.png'
111 #plt.savefig(filename)
112 print("Time for running ", N, "iteration :", time()-start, "seconds")
113 plt.show()
```

Bibliography

[1] Hupp, E.; Roy, S.; Watzke, M. (August 12, 2006). "NASA Finds Direct Proof of Dark Matter". NASA. Retrieved April 17, 2007.

[2] Bilek, Michal . 2016. Galaxy interactions: dark matter vs. Modified Newtonian dynamics (MOND).,arXiv:1601.01240v1 [astro-ph.GA] 6 Jan 2016.

[3] G. Gilmore, .1996. The Distribution of dark Matter in The Milky Way Galaxy.,Institute of Astronomy, Madingley Rd, Cambridge CB3 0HA, UK.[arXiv:astro-ph/9702081 [astro-ph]]. 1996.

[4] Frommert, Hartmut; Kronberg, Christine (August 26, 2005). "Classification of the Milky Way Galaxy". SEDS. Archived from the original on May 31, 2015. Retrieved May 30, 2015.

[5] Freeman, David., 2018., 'The Milky Way galaxy may be much bigger than we thought', *NBC Nes*, May 25, 2018, 3:28 PM WIB / Updated May 25, 2018, 6:40 PM, accessed July 14 from Kanazawa.

[6] BÜHRKE, Thomas. *Archaeology of the Milky Way.* PHYSICS & ASTRONOMY Galaxy. Article in The Astronomical magazine. Max Planck Research· January 2016.

[7] M. Milgrom, MOND-theoretical aspects, New Astron. Rev. **46**, 741-753 (2002) doi:10.1016/S1387-6473(02)00243-9 [arXiv:astro-ph/0207231 [astro-ph]].

[8] Sanders, R.H,. Modified Newtonian Dynamics: A Falsification of Cold Dark Matter., Hindawi Publishing Corporation Advances in Astronomy Volume 2009, Article ID 752439, 9 pages doi:10.1155/2009/752439.

[9] Dyer, Charles and Peter 1993 Softening in N-body simulation of collisionless systems ApJ. 409 67–60.

[10] Bartošková, Kateřina & Jungwiert, Bruno & Ebrova, Ivana & Jilkova, Lucie & Krizek, M.. (2011). Simulations of shell galaxies with GADGET-2: Multi-generation shell systems. Environment and the Formation of Galaxies: 30 Years Later. 10.1007/978-3-642-20285-8-39.

[11] F. Combes, 2014., Bulge formation in disk galaxies with MOND, Astron. Astrophys. **571**, A82 (2014) doi:10.1051/0004-6361/201424990 [arXiv:1409.4218 [astro-ph.GA]].

[12] G. N. Candlish, R. Smith and M. Fellhauer, Numerical simulations of Modified Newtonian Dynamics, J. Phys. Conf. Ser. **720**, no.1, 012012 (2016) doi:10.1088/1742-6596/720/1/012012.