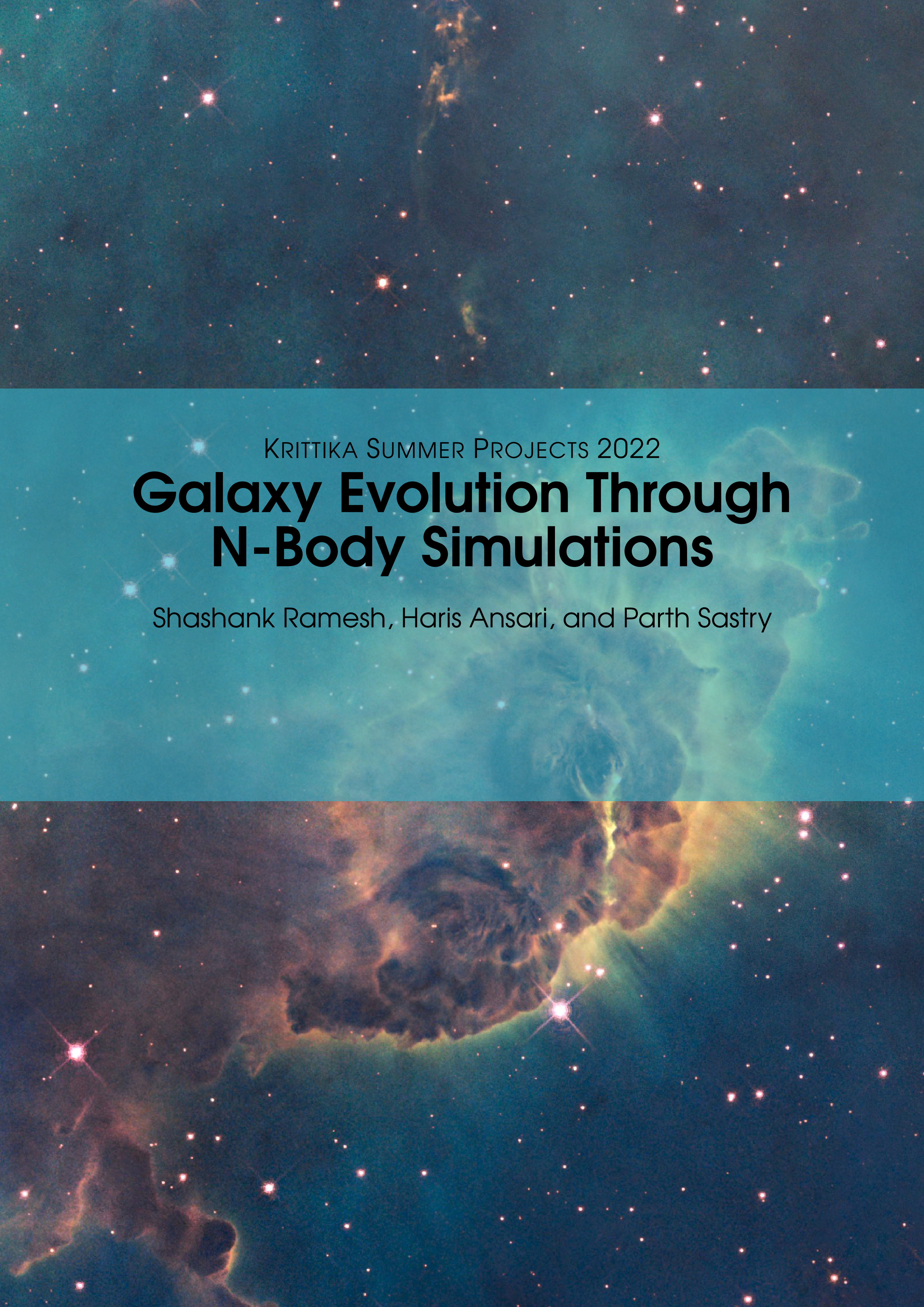


Krittika Summer Projects 2022

Galaxy Evolution Through N-Body Simulations

Shashank Ramesh, Haris Ansari, and Parth Sastry

KRITTIKA SUMMER PROJECTS 2022

Galaxy Evolution Through N-Body Simulations

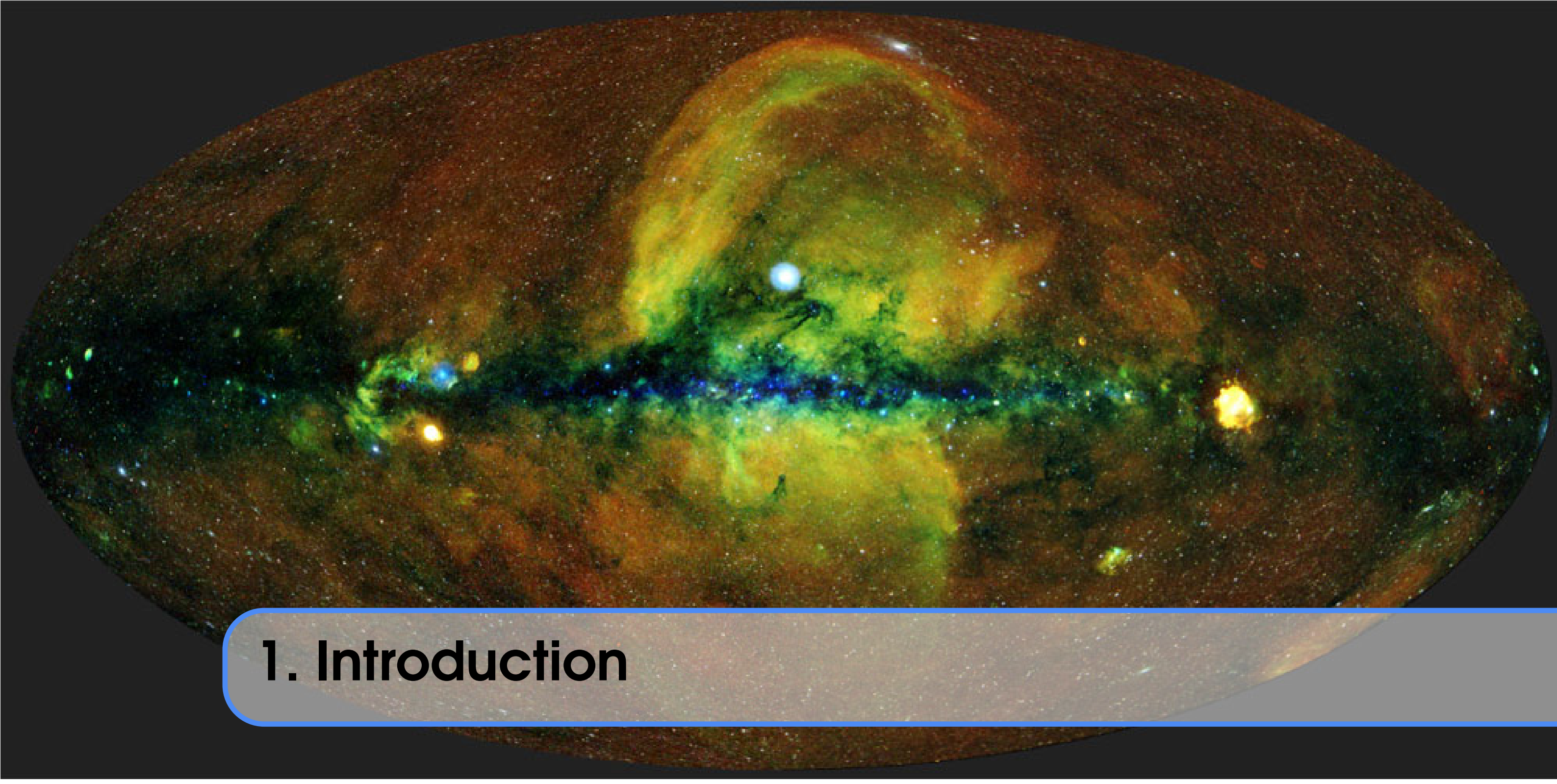
Shashank Ramesh[1], Haris Ansari[2], and Parth Sastry[3]

[1]Indian institute of Technology Madras, Chennai, 600036, India
[2]University of Hyderabad, Hyderabad, 500046, India
[3]Indian Institute of Technology, Bombay, Mumbai, 400076, India

Abstract

This project focuses on the simulation of astrophysical systems under the influence of a gravitational potential using the Barnes-Hut algorithm and the implementation of algorithms for outputting initial conditions for various astrophysical structures. This report gives basic details about the astrophysical structures in the universe and the currently understood mechanisms of galaxy formation and evolution. We then move on to methods to simulate systems involving a gravitational potential and talk about the Barnes-Hut algorithm, the driving force behind the simulations in this project. We then detail various models implemented to obtain initial conditions for the simulated systems, including Plummer model and Dehnen density and potential pairs. The report ends with an analysis of the notable features of the simulated systems.

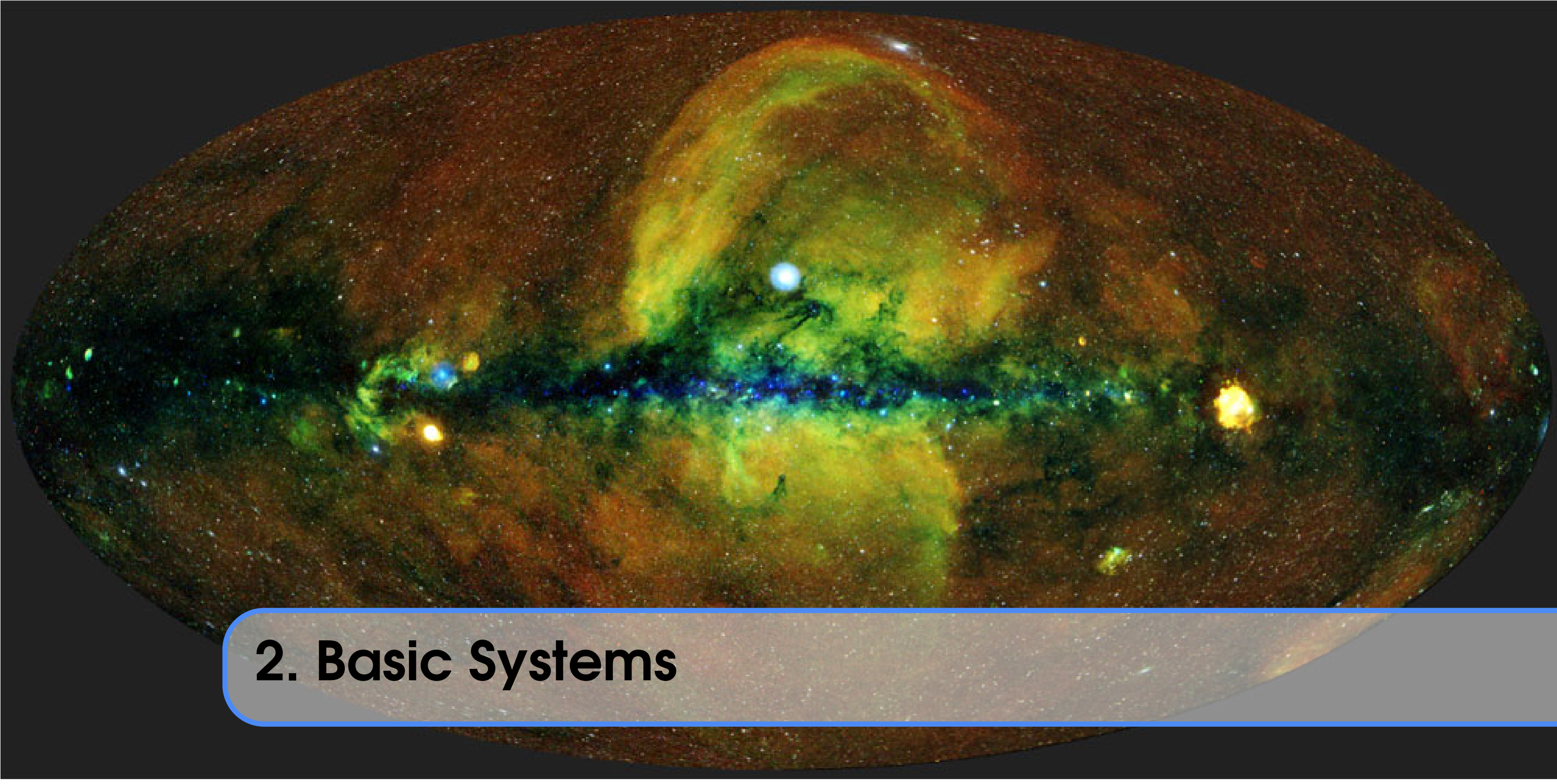
Contents



1. Introduction

The earliest galaxies in the universe took shape as little as 1 billion years into their lifetime. Two main ideas are given the most weight when discussing galaxy formation: The inward collapse of giant clouds of dust and gas under its own gravitational pull and the merging of small "lumps" of matter that resulted from density fluctuations in the early Universe. Newer research into this field focuses on topics like spiral arm formation and the effect of supermassive black holes on the galaxy's shape, to mention a few. In the astrophysical context this is particularly important for understanding galaxy interactions which are observed in the far and nearby universe

In this project, the main focus is on studying the evolution of galaxies and other astrophysical structures, considering only the effect of the gravitational force between the particles of the universe. Since gravitational forces constitute a significant component of the internal dynamics of the galaxy, we hope to see a decent reproduction of a galaxy's evolution with time. The study of the time evolution of galaxies allows us to trace back observational data to unravel the behaviour of galaxies at different scales and under different conditions. Data from telescopes like SKA, JWST, and Spitzer is being used to analyze the structure of galaxies. Galaxy evolution is also critical to understand how matter was distributed in the early universe.



2. Basic Systems

2.1 Newton's Law of Gravitation

- The equation for computing the force caused by the rest of the particles on a single particle can be computed by the law of gravity:

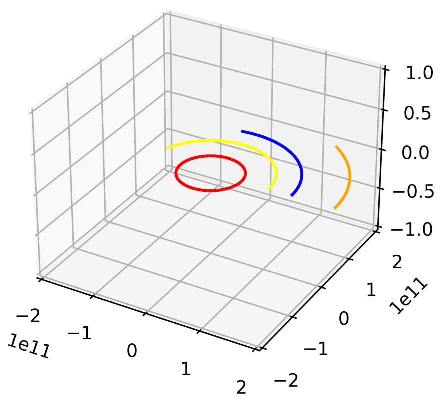
$$F_{ij} = m_i \ddot{r}_i = G \sum_{i=1, j\neq 1}^{N} m_i m_j \frac{(\vec{r}_i - \vec{r}_j}{|\vec{r}_i - \vec{r}_j|^3} \quad (2.1)$$

With-:
  - F_{ij} = Force acting between particles i and j
  - G = Gravitational constant
  - $m_i, m_j =$ Mass of the particle i and j
  - $\vec{r}_i - \vec{r}_j =$ Position of the particle i and j

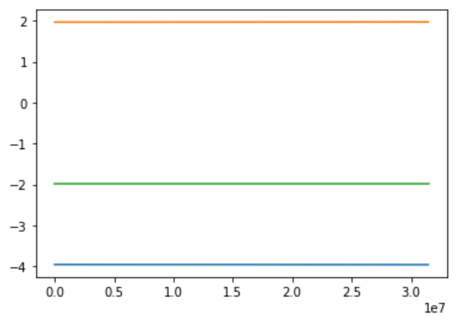
2.2 Solar System

We start with a well-known system: our very own Solar System. The solar system consists of the Sun, eight major planets, many minor planets, and other smaller structures. We pulled data on the Sun and eight significant planets and introduced it into Python. We then write a simple brute-force calculation to evolve the system, which is just calculating the force between objects pair by pair and updating the velocities and positions of the objects at each time step.

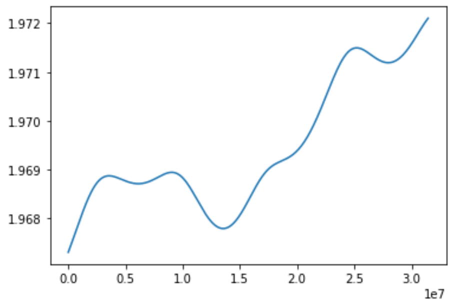
Figure 2.1 shows the orbits of the four inner planets, i.e. Mercury, Venus, Earth and Mars. The orbits are ellipses as expected, and the orbits are closed to a large degree. The precession expected due to the influence of the other planets shows up over a considerable simulation length. Energy conservation is obeyed quite well, with the kinetic and potential energy of the system following virial theorem to



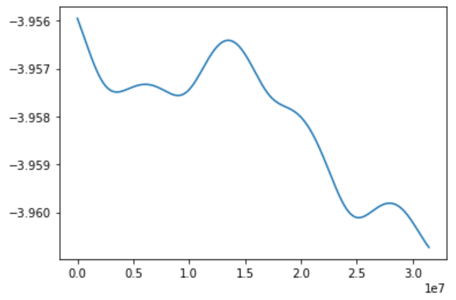
(a) Solar System



(b) Energies



(c) Kinetic Energy



(d) Potential Energy.

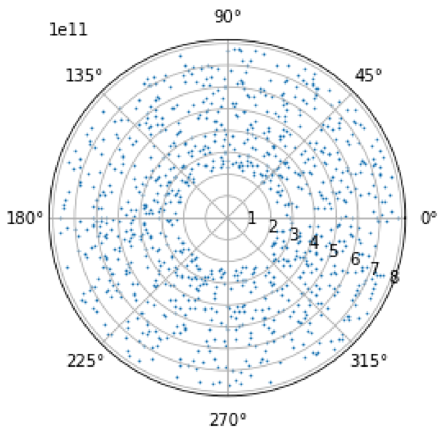
a large degree of accuracy (Figure 2.2). This is a great starting point to increase the number of particles and see the effect it has on the dynamics of the system and the time taken to simulate the system.

2.3 Asteroid Belt

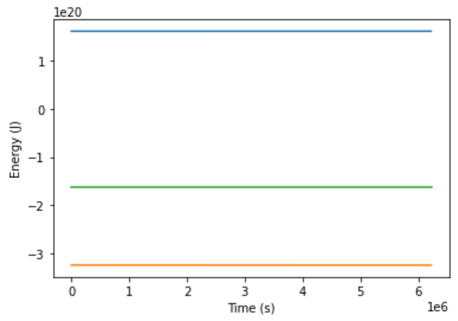
The next system we are using to illustrate such calculations is a basic mock up of an asteroid belt existing between Mars and Jupiter. The asteroid belt is a torus-shaped region in the Solar System, located roughly between the orbits of the planets Jupiter and Mars. It contains a great many solid, irregularly shaped bodies, of many sizes, but much smaller than planets, called asteroids or minor planets. This asteroid belt is also called the main asteroid belt or main belt to distinguish it from other asteroid populations in the Solar System such as near-Earth asteroids and trojan asteroids.

The asteroid belt is the smallest and innermost known circumstellar disc in the Solar System. About half its mass is contained in the four largest asteroids: Ceres, Vesta, Pallas, and Hygiea. The total mass of the asteroid belt is about 4% that of the Moon.

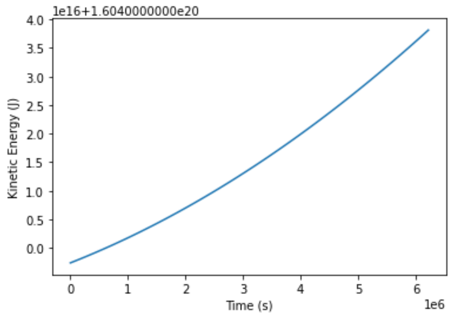
To initialize the system, the asteroids were given a random position somewhere between Mars and Jupiter by assigning them a radius and an angle in a plane polar coordinate system. The potential energy of every new asteroid was calculated with reference to all objects already in the system and a velocity was assigned based on virial theorem considerations. A mass was also randomly assigned to



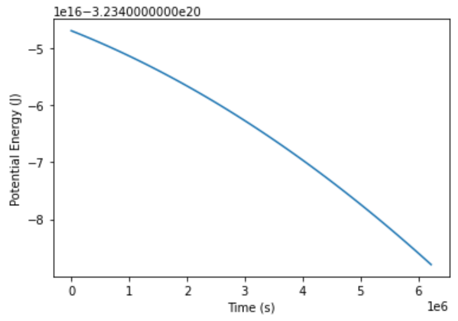
(a) Asteriod Belt mockup



(b) Energies



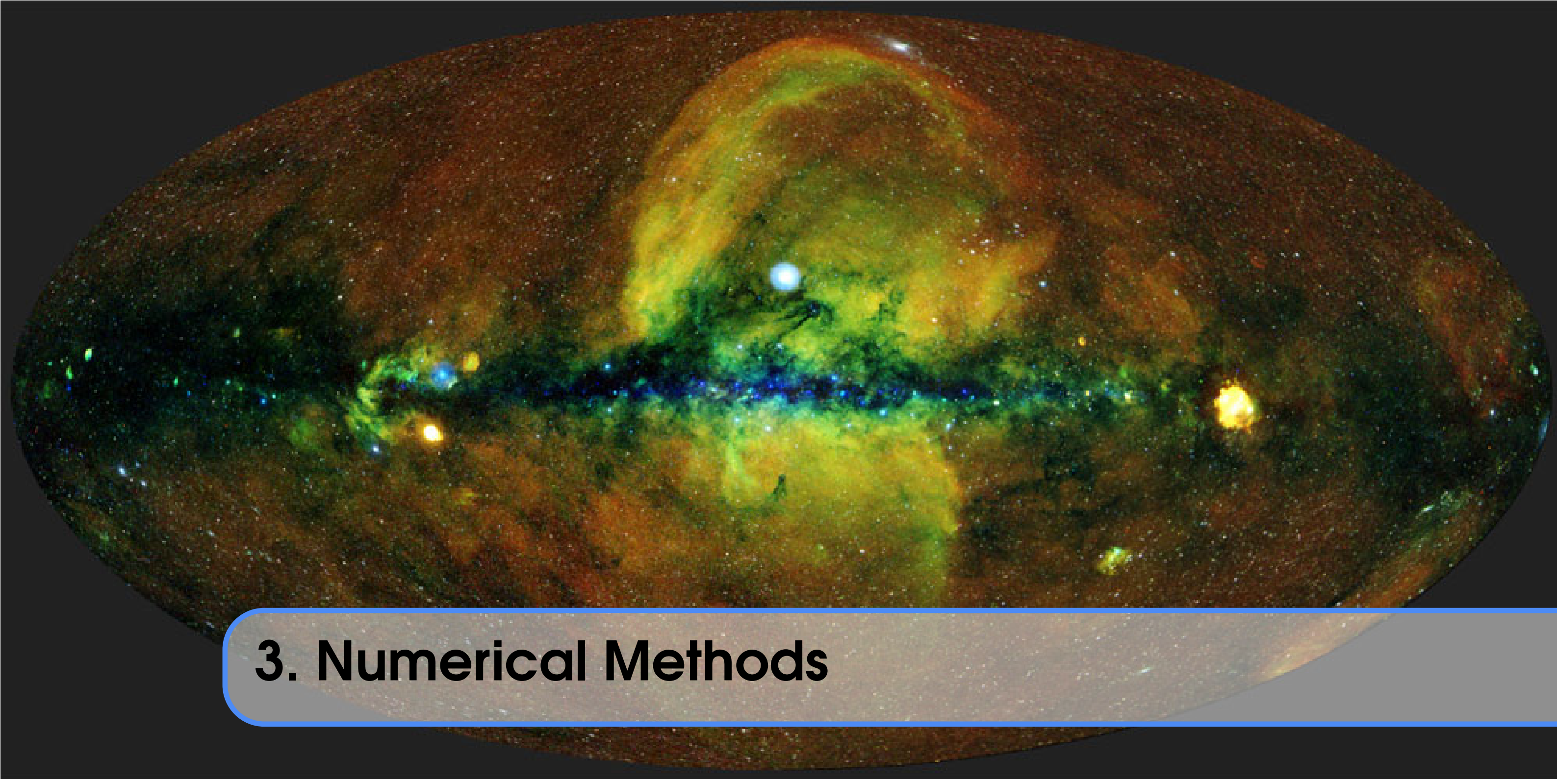
(c) Kinetic Energy



(d) Potential Energy.

each object based on data of the range of masses present in our asteroid belt. The graphs show a well behaved system where energy conservation is obeyed and the virial theorem is satisfied even as time evolves.

A very interesting phenomenon is seen if the initial conditions are not forced to follow the virial theorem. If the velocities and positions are distributed randomly, there is chaotic motion in the system. Particles will leave the asteroid belt at random when they are in a favourable position to do so, but this does not guarantee that the system will tend to go to a state where only particles with close orbits will remain significant in the system and equilibrium is restored.



3. Numerical Methods

3.1 Brute Force Method

In the Brute Force approach, we calculate all the possible solutions to find a satisfactory answer to a given problem. i.e. we will calculate the net force on mass m_i due to the rest of the mass distribution and iterating for each mass.

3.1.1 Algorithm

Initialization

The initial state of all Bodies is allocated to the three-dimensional fixed-length arrays:

```python
t0 = 0
dt = 86400; #time-interval
tf = 86400* 365 * 10
G = 6.67e-11 # Universal Gravitation Constant

t = arange(t0,tf+dt, dt)
r = zeros((len(t),3,9))
v = zeros((len(t),3,9))
m =array([1.989*10**6,0.33011,4.867,5.97,0.64,1898,568.34,86.81,102.41])
    *10**24
r[0,0,:]=array
    ([0,57.9,108.21,149.59,227.92,778.57,1453.53,2872.46,4495.06])*10**9
v[0,1,:]=array([0,47.36,35.02,29.78,24.07,13,9.68,6.8,5.43])*10**3
```

(language=Python)

Propagation

To seed the force of the simulation, an initial state is needed for the systems depending on gravitational or electric potential.Acceleration of particle is a result of summed force vectors, divided by the mass of the particle: $\vec{a} = \frac{\vec{F}}{m}$.
In basic propagation mechanisms, the Euler method is used below. The position

of an object at t_{n+1} is only dependent on its velocity at t_n, as the shift in position is calculated via: $\vec{r}_{t_{n+1}} = \vec{r}_{t_n} + \vec{v}_{t_n}$

```
for i in range(len(t)-1):#Time-Frame
  for j in range(1,5):# 4 = No. of Planets
    a = np.zeros(3) # acceleration vector
    for k in range(5):
      if k!= j:
        Rmag = np.linalg.norm(r[i,:,j]-r[i,:,k])# Magnitude of the distance
     between jth and kth object
        a[:] += -((G*m[k])/(Rmag)**2)* (r[i,:,j]-r[i,:,k])/(Rmag)
    v[i+1,:,j] = v[i,:,j] + a[:]*dt
    r[i+1,:,j] = r[i,:,j] + v[i+1,:,j]*dt
```

- The algorithmic complexity of solving an N-body system through a brute-force approach is $\frac{N(N-1)}{2} \approx O(N^2)$, making it impossible to simulate a system with more than a million objects, even on a high-end supercomputer.

3.2 Barnes-Hut Algorithm

- It uses a tree-based approximation scheme which reduces the problem's computational complexity from $O(N^2)$ to $O(NlogN)$.

- The crucial idea in speeding up the brute force n-body algorithm is to group nearby bodies and approximate them as a single body.

- This approximation is valid as long as the distance(d) from a point group to a particle is large and the radius of the group(r) is small to the distance between the group and the particle. The ratio of group distance (d) and group radius (r) is called the Multipole-Acceptance-Criterion (MAC):

$$\theta = \frac{r}{d} \tag{3.1}$$

  We will always use θ = 0.5, a value commonly used in practice. Note that if θ = 0, no internal node is treated as a single body, and the algorithm degenerates to brute force.
- If the group is sufficiently far away, we can approximate its gravitational effects using its centre of mass. Formally, if two bodies have positions (x_i, y_i) and (x_i, y_i), and masses m_i and m_j, then their total mass and centre of mass (R_x, R_y) are given by-:

$$M = m_i + m_j \tag{3.2}$$

$$R_x = (m_i x_i + m_j x_j)/m \tag{3.3}$$

$$R_y = (m_i y_i + m_j y_j)/m \tag{3.4}$$

3.2.1 Algorithm

Constructing the Barnes-Hut Tree

1. The algorithm starts by subdividing the domain into quadrants. Based on their location, We will refer to these quadrants by the following names: NW, NO, SW, and SO.
2. To add a particle to the tree, its quadrant is determined. If another particle is in the same quadrant, the quadrant is subdivided again.
3. This is repeated until each particle is located in its quadrant (however small this quadrant may be)
4. The InsertParticle algorithm must be run for every particle in the simulation. The following pseudocode can describe it:

```
    InsertParticle(Particle,Node):
        if(Particle.position is not in node):
            end function # if particle is not in node's bounding box return
        else:
            if (node has no particle):
                Node.Particle = Particle
            if(node is an internal node): #(that is, it has child nodes)
updateCenterOfMass(Node) # update center of mass of node
updateTotalMass(Node) # update total mass of node for child
            else:
                if Node.Child is None:
                    Node.Create_child()
                    Node.particle() = append.Particle
                    for particles in Node.Particle():
                        Insert(particles,Node.Child)
                else:
                    for Child in Node.Child(): #All childs of a Parent Node
                        InsertParticle(Particle,Child)
```

Calculating the Net Force

The "Dynamics" Algorithm takes in the particle we want to calculate the force for, the node we are currently comparing against, and θ(the opening angle).
To calculate the net force acting on body b, use the following recursive procedure, starting with the root of the quad-tree:

1. If the current node is external, calculate the force exerted by the current node on the given particle and add this amount to the net force.
2. Otherwise, calculate the ratio $\frac{r}{d}$. If $\frac{r}{d} < \theta$, treat this internal node as a single body, calculate the force it exerts on the particle, and add this amount to the net force on the given Particle.
3. Otherwise, run the procedure recursively on each of the current node's children.
4. The Dynamics algorithm must be run for every particle in the simulation. The following pseudocode can describe it:

```
Function Dynamics(Particle, Node, ThetaTolerance):
    if(Node.Children) is None:  # directly sum this
    node’s particle’s force on the particle we’re considering
        return Net_Force(Node,Particle)
    else:
        Theta = Node.Size/Distance(Node.Centre_of_Mass, Particle.Position)
        if Theta is less or equal to ThetaTolerance :
        # far field approximation is valid
            return farFieldForceCalc(Node.mass, Node.Centre_of_Mass, particle)
        else:
            for Child in Node.Children:
                Dynamics(Particle, Child,ThetaTolerance)
```

3.3 Head to head

The comparison between the new Barnes Hut algorithm (orange curve) and our old brute force method (blue curve) is striking. For tiny particle numbers, the brute force method is better. But as soon as the particle numbers cross 200, the Barnes Hut method takes over convincingly. Even at these smaller particle numbers, the brute force method showed an energy discrepancy of about 1%, whereas the Barnes-Hut process showed less than 0.01%. It is a far superior method for simulating systems with many particles.

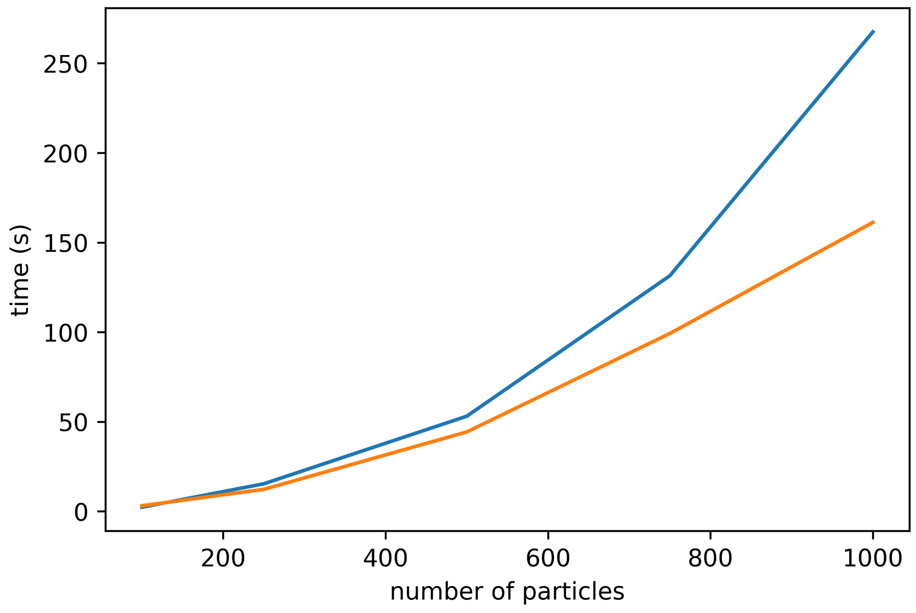
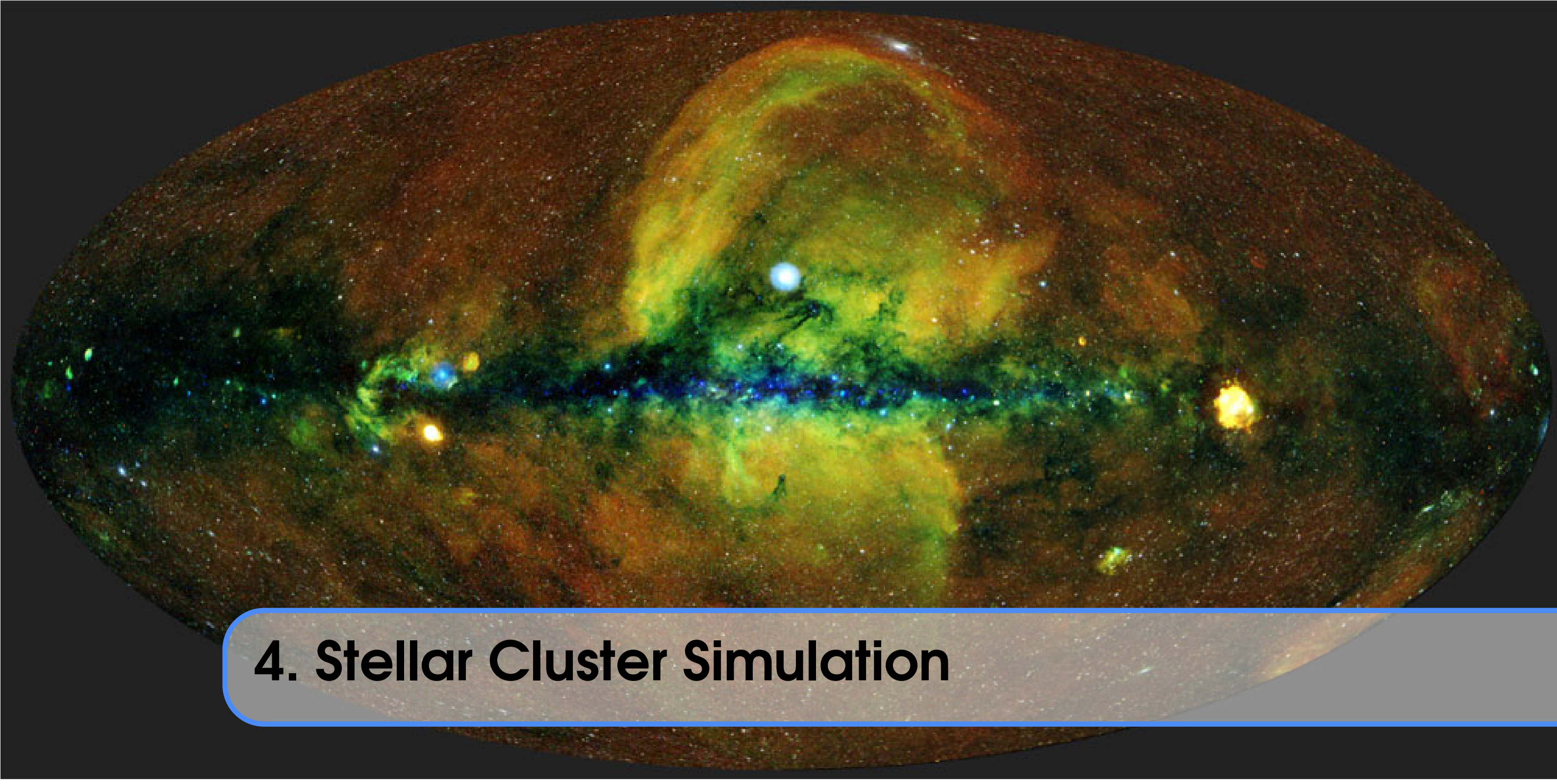


Figure 3.1: Comparison: Barnes Hut (orange) vs Brute Force (blue)



4. Stellar Cluster Simulation

4.1 Stellar Clusters

A star cluster is a group of stars that share a common origin and are gravitationally bound for some time. They benefit astronomers by providing a way to study and model stellar evolution and ages. The two basic categories of stellar clusters are open clusters, also known as galactic clusters, and globular clusters. Star clusters are significant because they allow astronomers to check models of stellar evolution and the ages of stars.

We will be using the Plummer model, a density law that H. C. Plummer first used to fit observations of globular clusters. It is now often used as a toy model in N-body simulations of stellar systems. We have implemented the Plummer model in Python and obtained initial conditions for a globular cluster of mass 10^{20} kg. The scale parameter a has been set to 10^{10} m, and the number of stars is 1000.

4.2 Plummer Model

All particles are given an equal mass, i.e. the total mass divided by the number of particles, to keep it simple.
The Plummer model uses the Schuster softened potential:

$$\Phi(r) = -GM\frac{1}{(r^2+a^2)^{\frac{1}{2}}} \tag{4.1}$$

Using Poisson's equation, we obtain the expression for the density distribution:

$$\rho(r) = \frac{M}{4\pi}\frac{3a^2}{(r^2+a^2)^{\frac{5}{2}}} \tag{4.2}$$

Now, the mass distribution can be obtained by integrating the expression for density over concentric shells. This will give us:

$$m(r) = M\left(1 + \frac{a^2}{r^2}\right)^{\frac{-3}{2}} \tag{4.3}$$

Inverting this equation gives us the dependence of radius on cumulative mass:

$$r(m) = \frac{a}{\sqrt{(\frac{M}{m})^{\frac{2}{3}} - 1}} \tag{4.4}$$

Similarly, using energy conservation and the expression for the potential, we arrive at an equation expressing the max. velocity of a particle at radius r:

$$v_e(r) = \sqrt{2GM}(r^2 + a^2)^{\frac{-1}{4}} \tag{4.5}$$

Using this expression and the distribution for the energy of the particles, we implement a rejection technique to assign velocities to each particle in the cluster. This completes our set of required initial conditions.

4.3 Results

The simulation shows in great detail the buzzing hub of activity that is a stellar cluster. In this system, energy is conserved to within 1%, and the virial theorem is obeyed as the system evolves. Since this velocity is inversely proportional to the scale parameter, stars move much slower in large clusters and this also leads to "mass leakage" from the cluster over time.

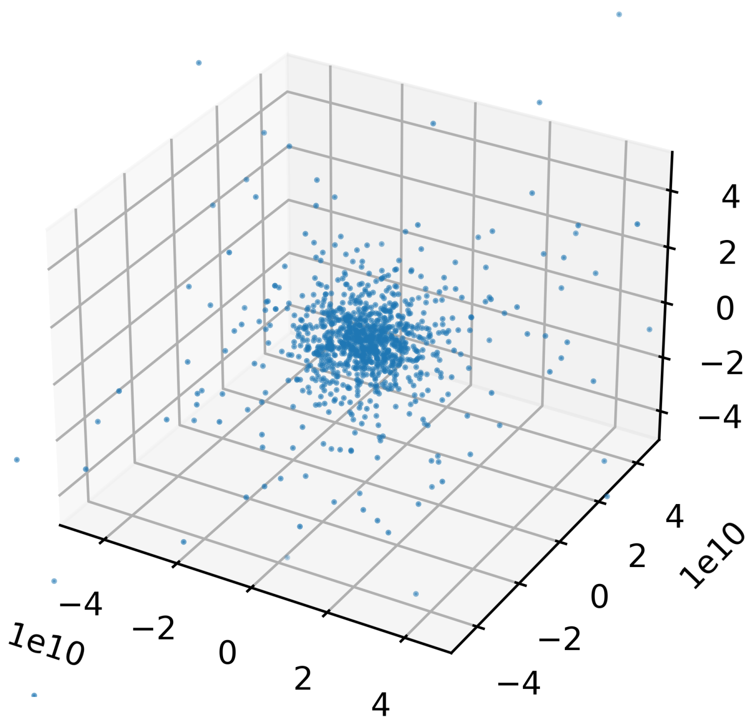


Figure 4.1: Stellar cluster initial setup

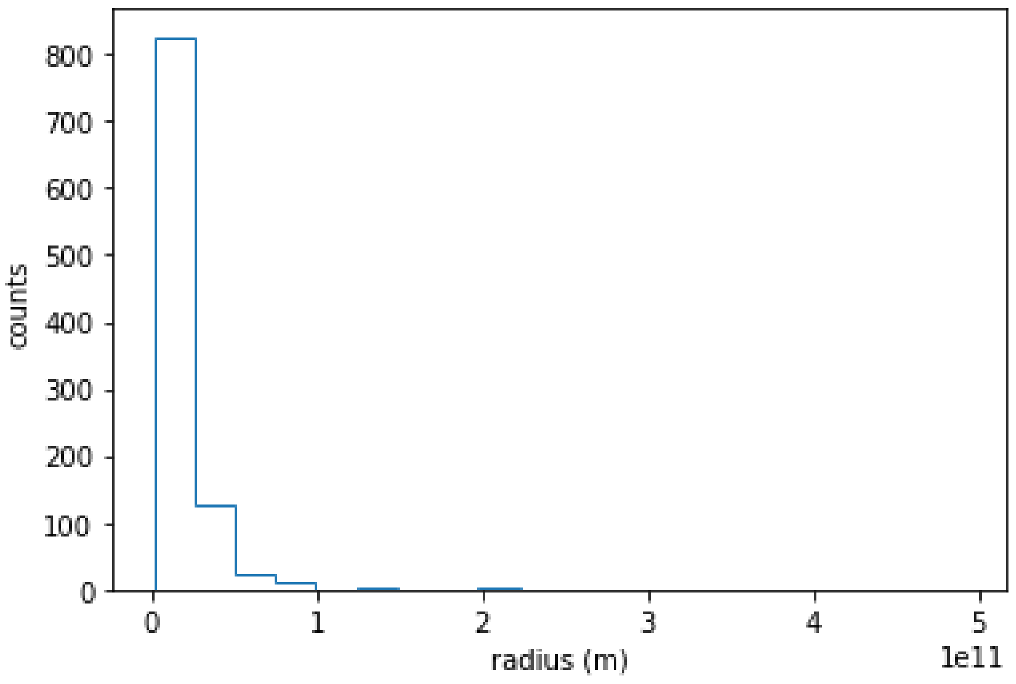


Figure 4.2: Radial distribution

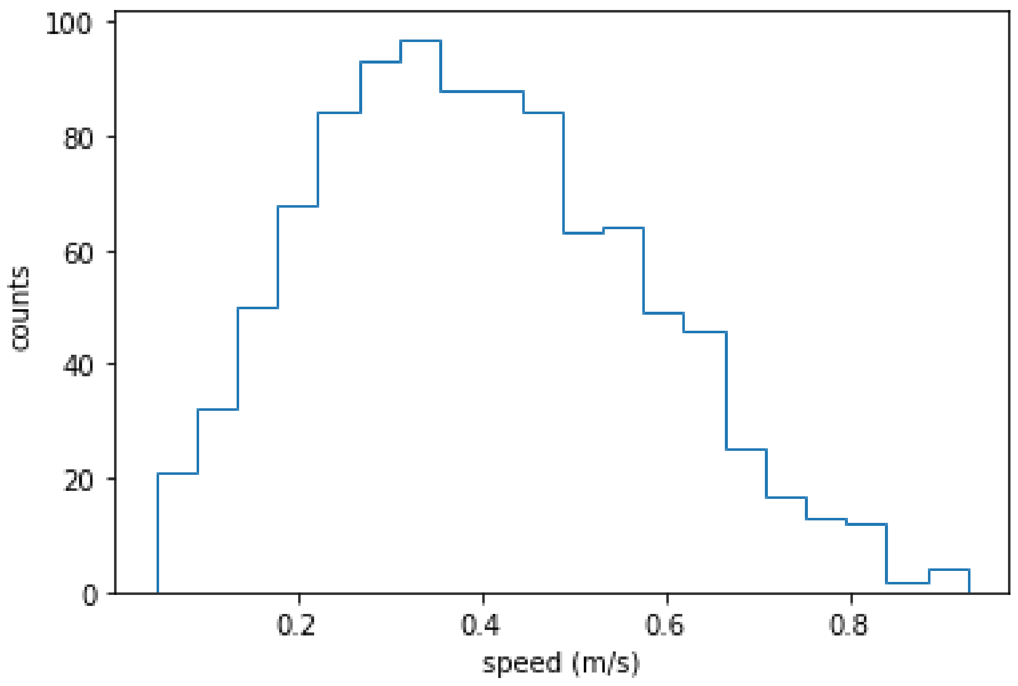


Figure 4.3: Velocity distribution

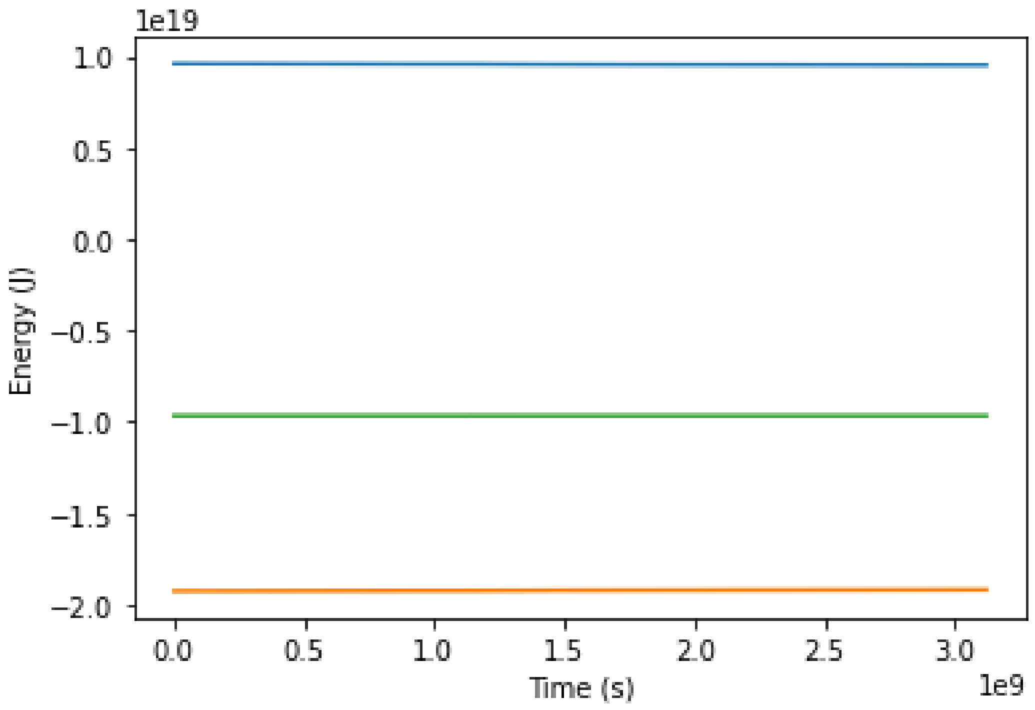
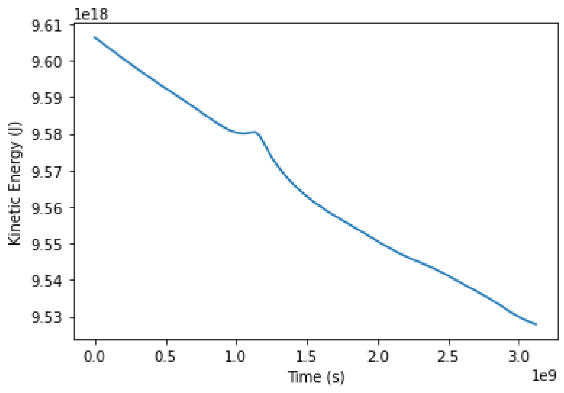
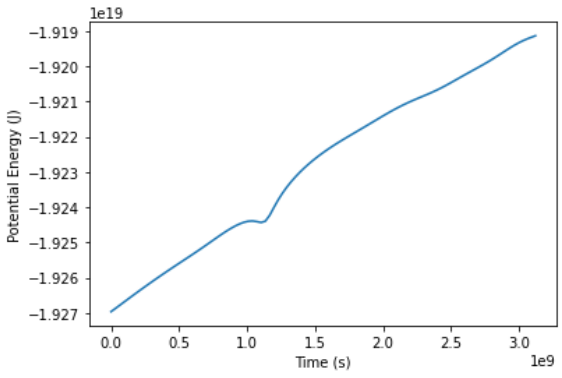


Figure 4.4: Energies



(a) Kinetic Energy



(b) Potential Energy.

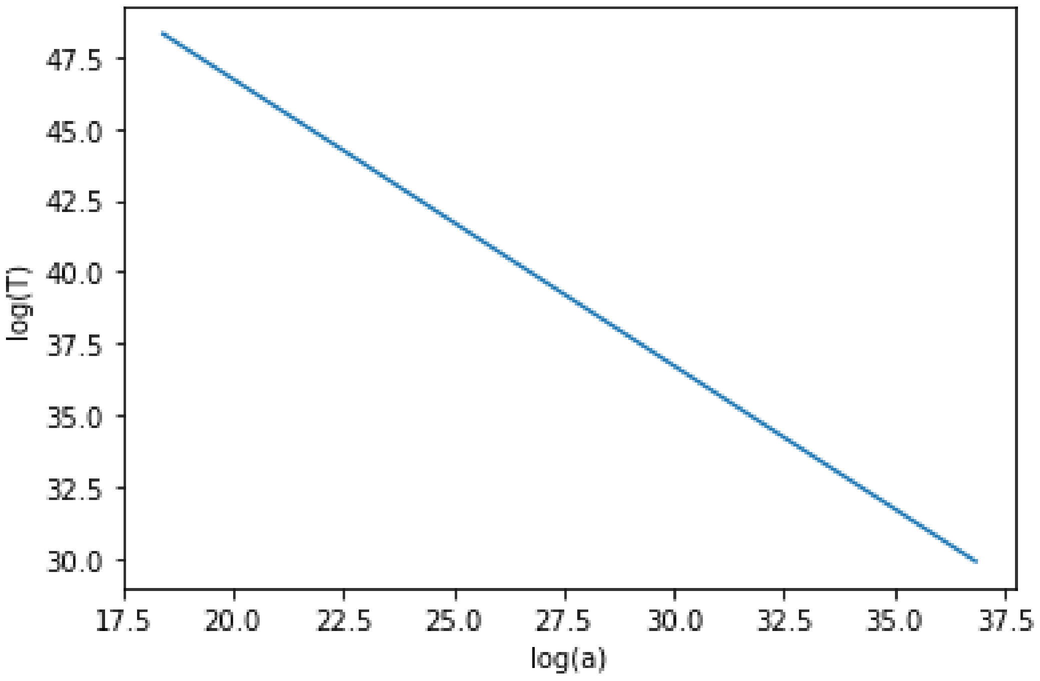
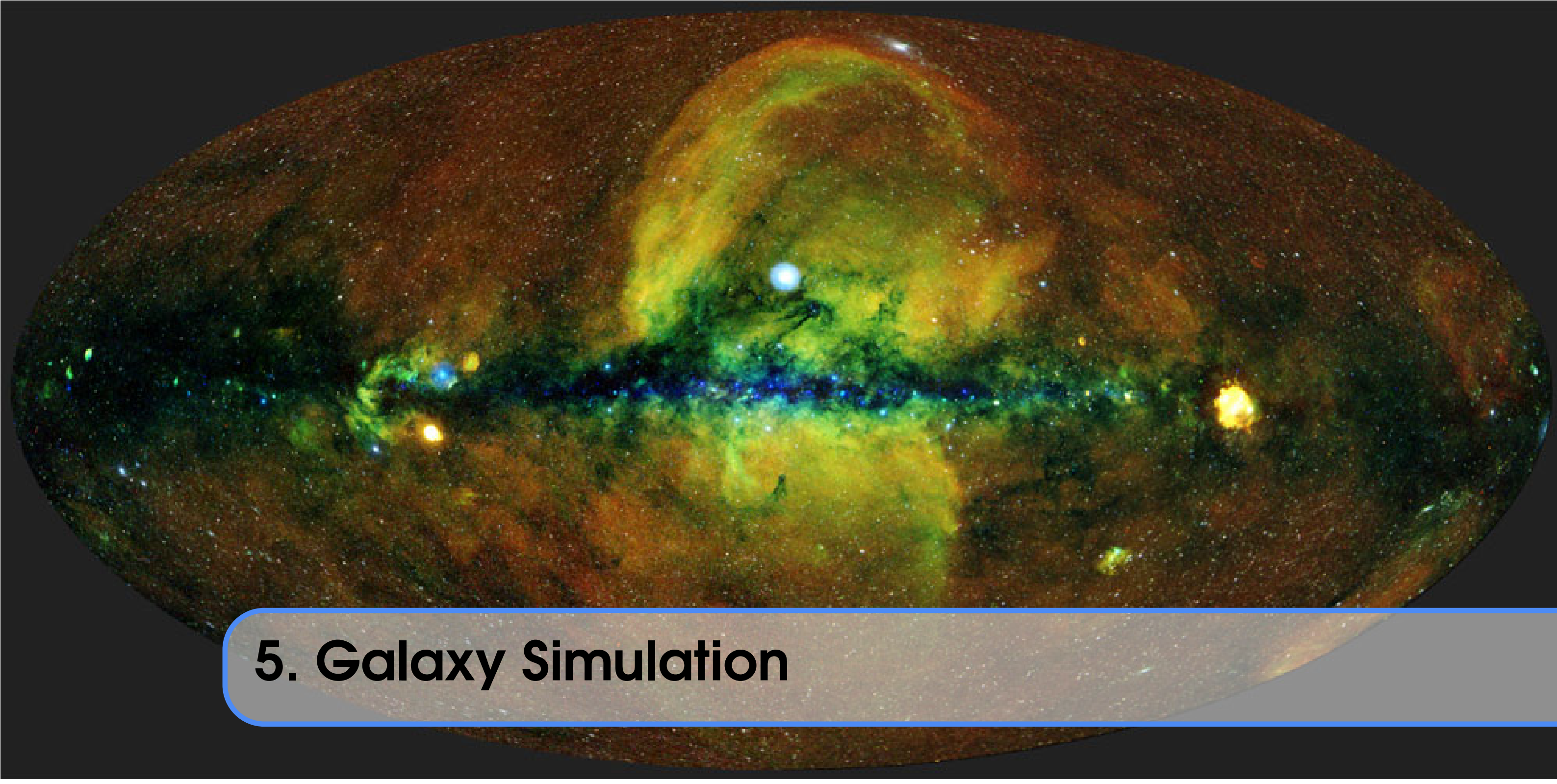


Figure 4.6: Kinetic energy vs scale parameter



5. Galaxy Simulation

5.1 Various profiles currently in use:

The observed luminosity profiles of ellipticals and bulges are often well described by the empirical formula $I \propto exp(-kR^{\frac{1}{4}})$ where k is a constant (de Vaucouleurs 1948). one is interested in simple models for the spatial stellar density, which in projection resemble de Vaucouleurs' profile in the outer parts, but not necessarily in the centre because the logarithmic slope of the luminosity profile vanishes at the center, which is in contradiction with observational data. Two important models are those introduced by Jaffe (1983) and Hernquist (1990), which have central stellar densities proportional to r^{-2} and r^{-1}, resulting in central surface densities proportional to R^{-1} and lnR^{-1}, respectively. These two models can be generalized to a family of density profiles with different central slopes, known as the Dehnen density and potential pairs (Dehnen 1993).

5.2 Dehnen pair:

The Dehnen distribution is a generalized distribution that returns different kinds of well-known potential and density pairs for different values of parameter γ.

$$\Phi(r) = \frac{GM}{a} \cdot \left\{ \begin{array}{ll} -\frac{1}{2-\gamma}[1-(\frac{r}{r+a})^{2-\gamma}], & \text{if } \gamma \neq 2 \\ ln\frac{r}{r+a}, & \text{if } \gamma = 2 \end{array} \right\} \tag{5.1}$$

$$\rho(r) = \frac{(3-\gamma)M}{4\pi} \frac{a}{r^{\gamma}(r+a)^{4-\gamma}} \tag{5.2}$$

Here, a is the scaling radius and M is the total mass of the system. $\gamma = 2$ gives the Jaffe model and $\gamma = 1$ gives the Hernquist model.

Using the prescription given in Springel et al (2005), velocities are assigned to the particles.Once again, the particles that make up any one component of the galaxy all have the same mass.

5.3 Implementation:

For this project, code was taken from the galstep initial condition generator and adapted to use for our purposes. The obtained initial conditions were sent to the driver code and the system was evolved. In terms of time step, the general rule of thumb is that if the velocity of a particle is v and its distance from another particle is d, then if the product of time step and v is on the order of d, the timestep is too high. We need a lower time step in that case.

A star moving with an average velocity v in a galaxy of radius R containing N equal-mass stars will have an aggregate velocity change of the order v in one relaxation time, t_{relax}, given by:

$$t_{relax} = \frac{N}{8lnN} \times \frac{R}{v} \tag{5.3}$$

In practice, we compare the relaxation time of a system with its crossing time, $t_{cross} = \frac{R}{v}$, to determine whether the star will suffer from significant encounters in a Hubble time. For the $N = 10^{11}$ stars in a typical galaxy with $R = 20$ kpc and $v = 200$ km/s, $t_{cross} = 10^8$ years, making $t_{relax} = 4.9 \times 10^{16}$ years. So, a typical star will not deviate at all over the period of a Hubble time. Such a system where the timescale for close encounters is much longer than a Hubble time, is called a collisionless system. Stars in the galaxy constitute such a collisionless system.

5.4 Results

The simulation showed good results in terms of energy conservation and the timestep chosen kept the energy within a reasonable bound near its original value. However, due to a limitation in computing power, we could not produce a good animation of the galaxy itself. Still, from the examples taken up earlier, we can keep faith that given enough processing power, this method can show the evolution of a galaxy to a very good degree. THe number of particles is 10000 and the timestep chosen was 250000 years. The results are detailed in the next pages:

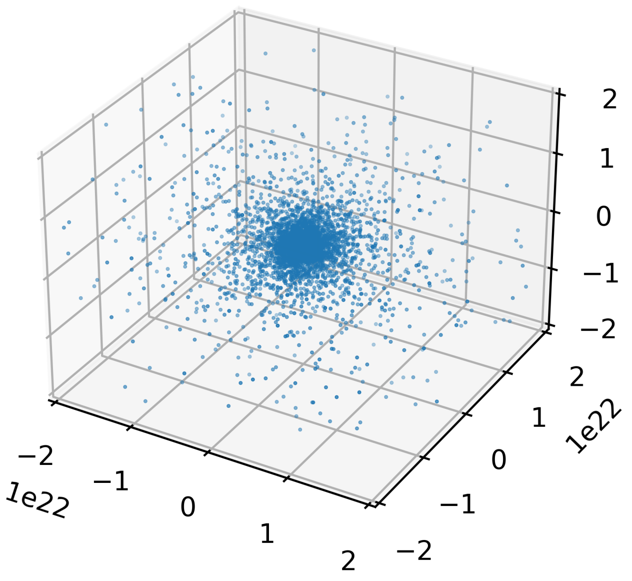
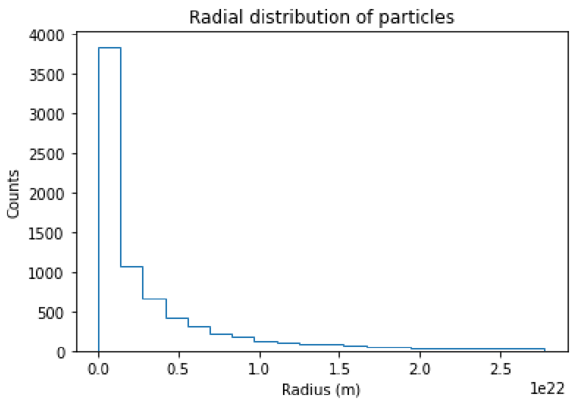
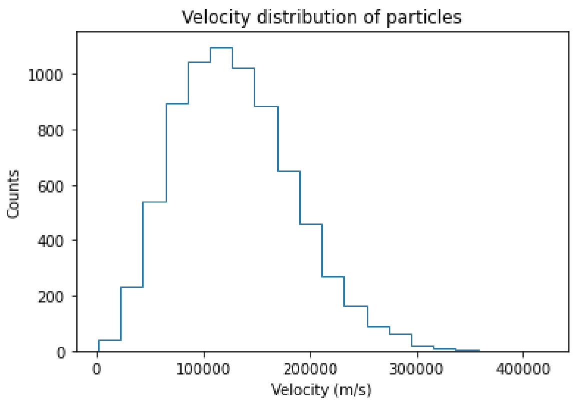


Figure 5.1: Galaxy initial setup



(a) Radial distribution of particles



(b) Velocity distribution of particles

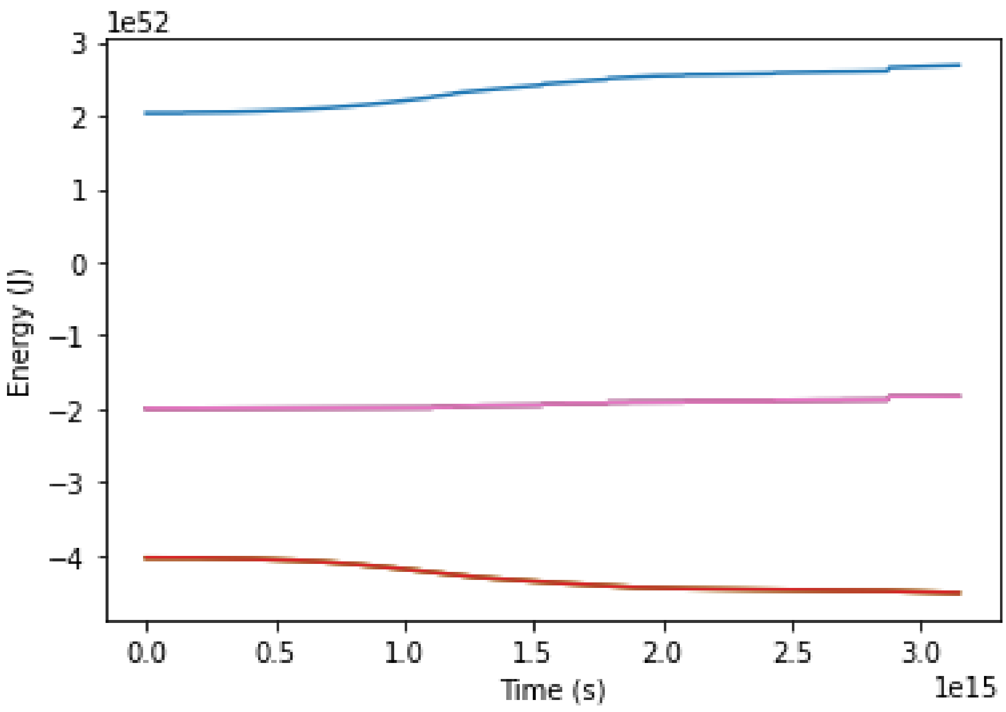
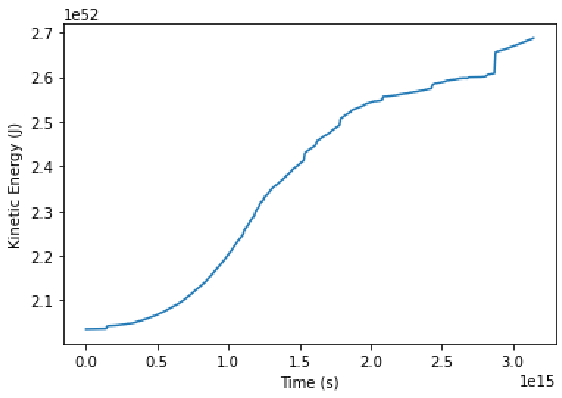
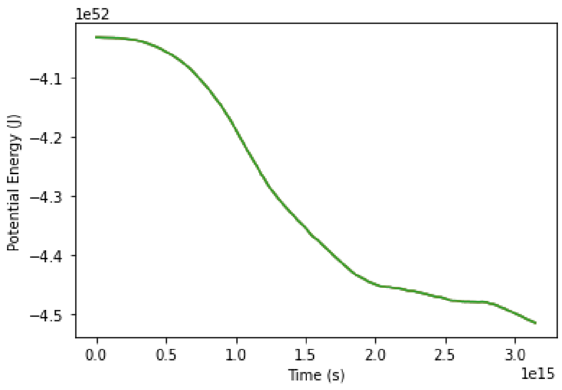


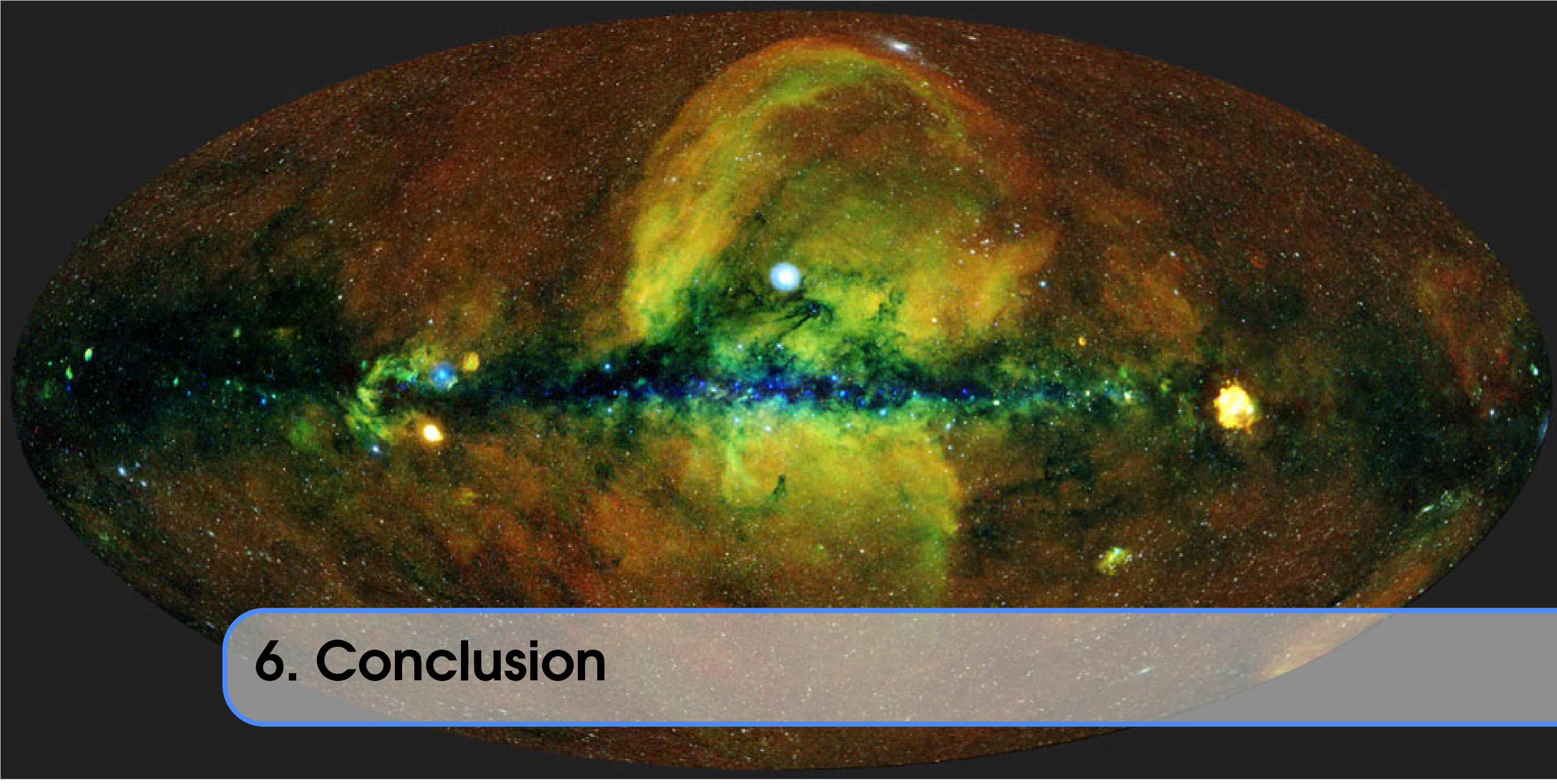
Figure 5.3: Energies



(a) Kinetic Energy



(b) Potential Energy.

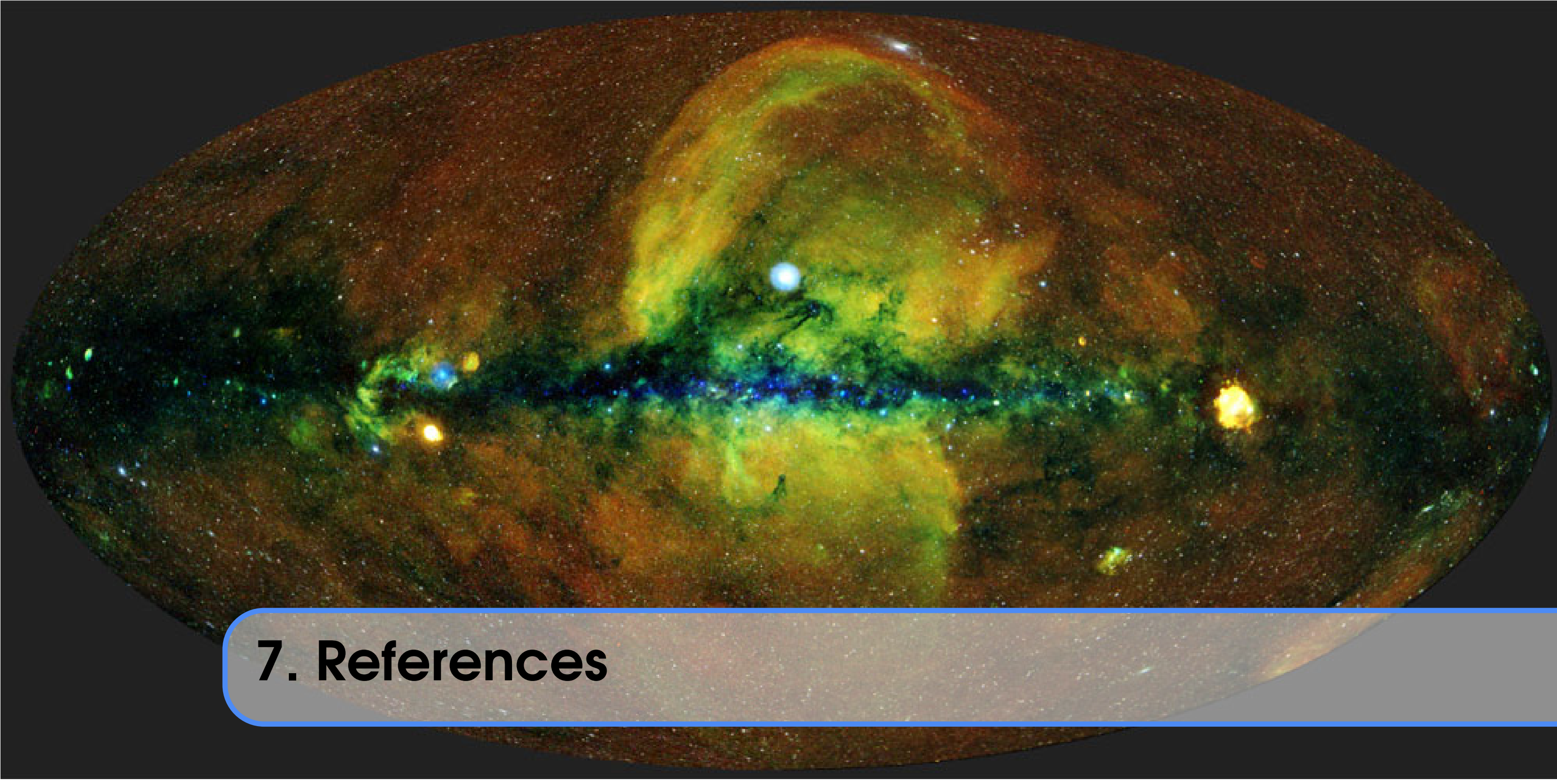


6. Conclusion

We have now seen and understood how complex gravitational systems such as galaxies behave over time and the properties that they satisfy. The simple systems gave us a basic idea of how gravitational forces are calculated and implemented to evolve a system with time. We also devised ways to check the accuracy of our simulation such as energy conservation, virial theorem verification and physical intuition combined with some observational data. We learnt various criteria for selecting a suitable time step for the simulation. We also compared two commonly-used methods for calculating gravitational forces and compared their efficiency and scalability. The Barnes-Hut algorithm is clearly the far superior choice for large particle numbers.

When a system is initialized in a state that obeys virial theorem, it will continue to obey virial theorem and the particles will move in closed orbits around the barycentre of the system. Systems that do not have this property, on the other hand, show more chaotic motion, with particles escaping the system randomly. Such systems do not tend to evolve in such a way that virial theorem is obeyed at any future instant in time, which has consequences for many-particle dynamical systems.

N-body simulations are a highly useful tool to analyze the motion of large systems of particles, with the method finding use in other fields of physics like plasma physics in different avatars. This project is a demonstration of how it can be implemented to study galaxies and other intricate systems with a good degree of accuracy. Due to restrictions on computing power, we could not simulate the larger systems for very long times, but we were able to use lengths just long enough to see things moving around and to verify the correctness of our method. With an increase in computing power, we can apply the techniques learnt here to do even better simulations to bring out more features of the structures we've analyzed in this project. Overall, this project was a highly enlightening and enthralling experience and we thank Krittika, the Astronomy club of IIT Bombay for this opportunity. We'd also like to thank our mentor Parth Sastry for his support and guidance.



7. References

- https://ui.adsabs.harvard.edu/abs/1979ApJ...228..664A/abstract
- https://ui.adsabs.harvard.edu/abs/2005MNRAS.361..776S/abstract
- https://github.com/ruggiero/galstep
- https://academic.oup.com/mnras/article/378/2/541/960760
- https://people.ast.cam.ac.uk/ puchwein/NumCosmo_lect_2016/ NumericalCosmology01.pdf
- https://www.usm.uni-muenchen.de/people/puls/lessons/numpraktnew/ nbody/nbody_manual.pdf
- https://astrocrash.net/files/2005%20James%20Guillochon.pdf
- https://articles.adsabs.harvard.edu/pdf/1974A%26A....37..183A
- https://www.ifa.hawaii.edu/ barnes/talks/vlasix15/barnes.pdf
- https://anaroxanapop.github.io/behalf/
- https://jheer.github.io/barnes-hut/
- https://academic.oup.com/mnras/article/265/1/250/975506