Universiteit
Leiden
The Netherlands

Chaotic Dynamics in N-body systems

Boekholt, T.C.N.

Citation
Boekholt, T. C. N. (2015, November 10). *Chaotic Dynamics in N-body systems*. Retrieved from https://hdl.handle.net/1887/36077

Version:	Not Applicable (or Unknown)
License:	Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden
Downloaded from:	https://hdl.handle.net/1887/36077

Note: To cite this publication please use the final published version (if applicable).

Cover Page

Universiteit Leiden

The handle http://hdl.handle.net/1887/36077 holds various files of this Leiden University dissertation

Author: Boekholt, Tjarda
Title: Chaotic dynamics in N-body systems
Issue Date: 2015-11-10

1

Introduction

This thesis focusses on the evolution of dynamical systems in the universe. The main systems of interest are planetary systems and star clusters. Understanding the structure and evolution of these systems includes calculating the orbits of the constituent bodies. This is rather complex due to the non-linear, chaotic nature of most orbits. Therefore, we need to resort to numerical N-body techniques to approximate the orbits using sophisticated software and modern hardware.

A central theme in this thesis is the growth of a small perturbation. This could be a small numerical error that propagates through the system causing numerical noise. This could bias results from N-body simulations, questioning their reliability. The initial perturbation could also be physical and its propagation determines the stability of an orbit.

This chapter provides a general introduction to the topics mentioned above, and subsequent work in this thesis on two new N-body algorithms, testing the reliability of N-body simulations and determining the origin of chaos in the orbit of Comet Halley.

1.1 DYNAMICAL SYSTEMS

In the universe there is an endless number of examples of dynamical systems. Any system that consists of a collection of objects or so-called bodies, which move around through space and time due to their mutual forces, is a valid candidate.

Our solar system is a familiar example consisting of several types of objects. We have the sun at the center (Copernicus, 1543), the planets revolving around the sun in ellipses (Kepler, 1609), moons orbiting the planets (Galilei, 1610), and many smaller objects such as asteroids, comets (Halley, 1705) and so on (see Fig. 1.1, top image). These objects all have one thing in common, they all have mass. Since the pioneering work by Newton (1687), we know that bodies with mass attract each other. Every object feels the gravity from the other objects. As a consequence, our solar system is not static, but evolves because all the bodies are moving. If all the bodies in the system are

moving, then it is also possible for two of them to experience a close encounter, like Halley's Comet and Earth back in 1910.

A different kind of dynamical system is a star cluster (see Fig. 1.1, bottom image). To first order, such a system consists solely of stars. The number of stars varies per cluster, from a few dozen to millions. The number of stars matters for the evolution of the system as a whole. Hypothetically speaking, if we put our sun in a star cluster (modern theories actually predict this was the case when our sun was younger (Martínez-Barbosa et al., 2015; Jílková et al., 2015)) and there are only a few other stars in the cluster, then the gravitational pull from each star on the sun would form a significant contribution. If instead, we have a million other stars, each individual contribution becomes very small and so the sun would follow a smoother orbit set by the overall, background potential. This dependency on the number of stars is usually captured by the relaxation time of the cluster (Chandrasekhar, 1942), which is the time scale on which the sun would significantly alter its orbit in its birth cluster.

Only in the dense cores of massive star clusters, do close encounters between two or more stars become important again. Three-body encounters especially, form a separate category of dynamical systems. They are important for the formation of binary stars. Initially three stars will encounter each other and exchange some energy. If one of the three stars manages to steal enough energy from the other two, it will fly away out of the core of the cluster. The remaining two stars have lost energy and form a bound two-body system, e.g. a binary star (Szebehely, 1972). This binary star will interact with the other surrounding stars, effectively inserting kinetic energy into the core of the cluster. Many theoretical and numerical studies have been performed to measure these detailed energy exchanges and its feedback on the cluster (e.g. Heggie, 1975; Hut & Bahcall, 1983; McMillan & Hut, 1996; Boekholt et al., 2015). For a more general background on star cluster dynamics, the following books are highly recommended: **?**, **?** and **?**.

Planetary systems, star clusters and three-body scatterings are just a few examples of dynamical systems in the universe. There are many more, such as galaxies consisting of billions of stars or galactic nuclei which include supermassive black holes. For all these dynamical systems there are several things we would like to know. Specifically, how many and what type of bodies are present in the system, how they are distributed throughout the system, how does the global structure evolve and what is its final fate. As an illustration, we take our own solar system again, but only consider the sun and the planets. We have

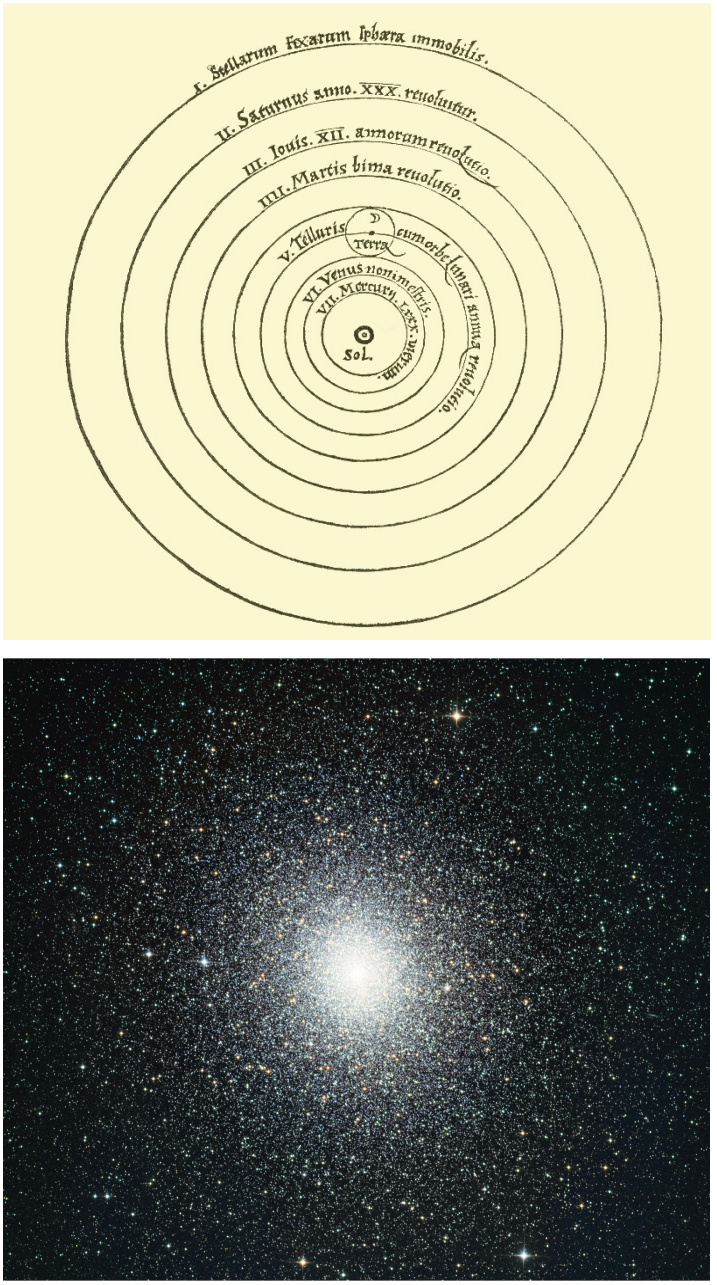

Figure 1.1: Two examples of dynamical systems: schematic overview of our solar system, including the sun, the planets and our moon (top image) and the star cluster 47Tuc, consisting of over a million stars (bottom image). Credit to Copernicus (1543) and Dieter Willasch (Astro-Cabinet).

measured their masses, positions and velocities as they are currently. One of the open problems is how these orbits evolve over billions of years. One possibility is that the solar system will always remain more or less the same, i.e. the system is stable (Ito & Tanikawa, 2002). Some studies have shown that it is also possible for orbits to change significantly, resulting in collisions between planets (Laskar & Gastineau, 2009) or an escape from the system altogether (Laskar, 2008). To investigate the origin and evolution of dynamical systems we need a mathematical model that captures the behaviour of bodies in motion.

1.2 THE N-BODY PROBLEM

Newton (1687) defined a mathematical model, called the N-body problem, which describes the following: we have a dynamical system consisting of N bodies (with $N = 1, 2, 3, ...$), each having a mass, position and a velocity. What are the positions and velocities at any time in the future or in the past? If the bodies would not influence each other, Newton's first law states that the body will continue moving on a straight trajectory with constant velocity. If the bodies do feel each other through a mutual force, they will experience an acceleration and generally follow curved trajectories. This relation is given by Newton's second law of motion

$$F = ma, \tag{1.1}$$

with F the force, m the mass of the body experiencing the force and a the acceleration. When the apple hit Newton's head, he realized that the attractive force that acts between all bodies with mass is gravity. The force of gravity is proportional to the masses of the two bodies and inversely proportional to the square of the distance between them

$$F = \frac{GMm}{r^2}. \tag{1.2}$$

Here G is the gravitational constant, M and m are the masses of the two interacting bodies and r the distance between them. To be more precise, the mass in Eq. 1.1 is the inertial mass, whereas the mass in Eq. 1.2 is the gravitational mass. Experiments have shown however, that they are equal up to at least 13 decimal places (Poisson & Will, 2014). We can therefore equate Eq. 1.1 and 1.2 and at the same time take the sum over all other bodies

$$\vec{a}_i = G \sum_{j \neq i} \frac{m_j}{r_{ij}^3} \vec{r}_{ij}. \tag{1.3}$$

Here $\vec{a}_i$ is the total acceleration experienced by particle i, due to all other particles with masses m_j at distances of $r_{ij} = |\vec{r}_j - \vec{r}_i|$. For every object we calculate its acceleration due to all the other objects. Next step is to use this acceleration to calculate the position and velocity at some time in the future. This is the tricky part.

The acceleration as a function of time is generally not a simple function, but rather complex and chaotic. Except for a few cases with $N = 2$ (Newton, 1687) and $N = 3$ (Euler, 1767; Lagrange, 1772) we can solve the orbits analytically. For the rest of the cases we need to measure the acceleration, time step by time step. This discretisation of the N-body problem introduces an error in the solutions, but makes it ideal for solving it on a computer.

1.3 N-BODY SIMULATIONS

1.3.1 Algorithms

Soon after the invention of the computer, did the first N-body simulations arise. Among the first ones to solve the N-body problem on a computer were von Hoerner (1960) and Aarseth (1963), with the aim of solving the orbits of stars under their mutual gravity.

Solving the N-body problem on a computer involves two main ingredients: an integration method and a time step criterion. The integration method determines how the new position and velocity are calculated from the current position, velocity and acceleration. The simplest integrator is the Euler method:

$$r\left(t + \Delta t\right) = r\left(t\right) + v\left(t\right)\Delta t, \tag{1.4}$$

$$v\left(t + \Delta t\right) = v\left(t\right) + a\left(t\right)\Delta t. \tag{1.5}$$

Here t is the current time, Δt is the time step size, r is the position and v the velocity. By iteratively performing the algorithm of applying Eq. 1.3, 1.4 and 1.5, we can sample the orbits of the bodies in time and study the evolution of dynamical systems.

The Euler algorithm is very simple and clear, but not very precise. If we regard it as a first order Taylor expansion, then the higher order terms are neglected which introduces truncation error in the results. One way to reduce this type of error is to select an appropriate time step criterion. In principle the time step size should be as small as possible, but this will increase the number of integration steps, which dramatically increases the amount of CPU time. Therefore, there is a trade off between the amount of precision and speed. A variety of time step criteria exist in the literature, two of which are

$$\Delta t = \mathsf{min}\left\{\frac{r_{ij}}{v_{ij}}\right\}, \tag{1.6}$$

$$\Delta t = \mathsf{min}\left\{\sqrt{\frac{r_{ij}}{a_{ij}}}\right\}, \tag{1.7}$$

The first criterion makes sure that the closest pair of bodies cannot collide within a single time step, because the time step size becomes smaller if there are close encounters. The second criterion is similar but also works for zero velocities. For more advanced time step criteria and also further details on N-body algorithms and simulations we refer the reader to Aarseth (2003) and The Art of Computational Science project by Hut and Makino[1].

Two commonly used N-body integrators are the Leapfrog and the Hermite method. The Leapfrog algorithm goes as follows

$$v(t + \frac{\Delta t}{2}) = v\,(t) + a\,(t)\,\frac{\Delta t}{2}, \tag{1.8}$$

$$r(t + \Delta t) = r\,(t) + v\,(t)\,\Delta t + \frac{1}{2}a\,(t)\,\Delta t^2, \tag{1.9}$$

$$v(t + \Delta t) = v(t + \frac{\Delta t}{2}) + a\,(t + \Delta t)\,\frac{\Delta t}{2}. \tag{1.10}$$

Compared to the Euler scheme, the new position is calculated using an extra term proportional to Δt^2. The main difference is that the velocity is updated in two steps. First, half of the time step is taken using the current acceleration, then the second half of the time step is taken using the new acceleration which is calculated at time $t+\Delta t$. The updated positions and accelerations in the second step of the velocity calculation, has the result that the energy error remains bound. This property is characteristic for symplectic methods, and is especially useful for integrations of planetary systems (**?**).

The Hermite integrator (**?**) is a fourth-order, non-symplectic method that is useful for accurate integration of collisional systems such as star clusters. The algorithm consists of a prediction step

$$r_p = r\,(t) + v\,(t)\,\Delta t + \frac{1}{2}a(t)\Delta t^2 + \frac{1}{6}j(t)\Delta t^3, \tag{1.11}$$

$$v_p = v\,(t) + a\,(t)\,\Delta t + \frac{1}{2}j(t)\Delta t^2, \tag{1.12}$$

[1] http://www.artcompsci.org/

and then a correction step that makes use of the initial coordinates and the predicted coordinates:

$$r(t+\Delta t) = r(t) + \frac{1}{2}\left(v(t)+v_p\right)\Delta t + \frac{1}{12}\left(a(t)-a_p\right)\Delta t^2, \qquad (1.13)$$

$$v(t+\Delta t) = v(t) + \frac{1}{2}\left(a(t)+a_p\right)\Delta t + \frac{1}{12}\left(j(t)-j_p\right)\Delta t^2. \qquad (1.14)$$

Here $j(t)$ is the jerk which is the time derivative of the acceleration, and a_p is the acceleration calculated using the predicted positions[1].

Once the software has been written it has to run on a piece of hardware. For simulations of a few particles a personal desktop is sufficient. A gain in speed can be obtained by using the fastest processors available on the market. For larger dynamical systems it becomes efficient to parallellise the computations. As was shown in the previous section, to calculate the acceleration of a single body, we need to iterate over all the other bodies, which is an operation of order N. However, we want to know the acceleration of each particle, which is another factor N, so that the total cost is of order N^2. Dividing the computational work over as many cores as possible can lead to an increase in speed. It is worth noting that there is a trade off between the time the computer spends calculating and the amount of time the computer cores are communicating and passing data. The challenge for N-body developers is to design a code which runs efficiently with as many cores as possible (Portegies Zwart et al., 2008).

Approximate methods exist which trade in some precision for better performance by relieving the constraint of having to consider all pairs of bodies. One familiar example is the tree method where clumps of bodies are replaced by a single, centre of mass body (**??**). Finally, calculations on Graphic Processing Units (GPUs), instead of Central Processing Units (CPUs), can decrease the duration of the simulations by one or two orders of magnitude. The current fastest N-body code on the planet is a tree method compatible with clusters of GPUs (**?**).

Since the N-body problem cannot be solved exactly, much effort has gone into looking for new, improved algorithms for solving the N-body problem. Different methods can be more appropriate for different problems. For example, some might aim for speed in order to be able to handle large simulations of globular clusters and galaxies, whereas others might aim for high precision for planetary and few-body systems. The quest for the ultimate N-body method that is fast and precise still continues.

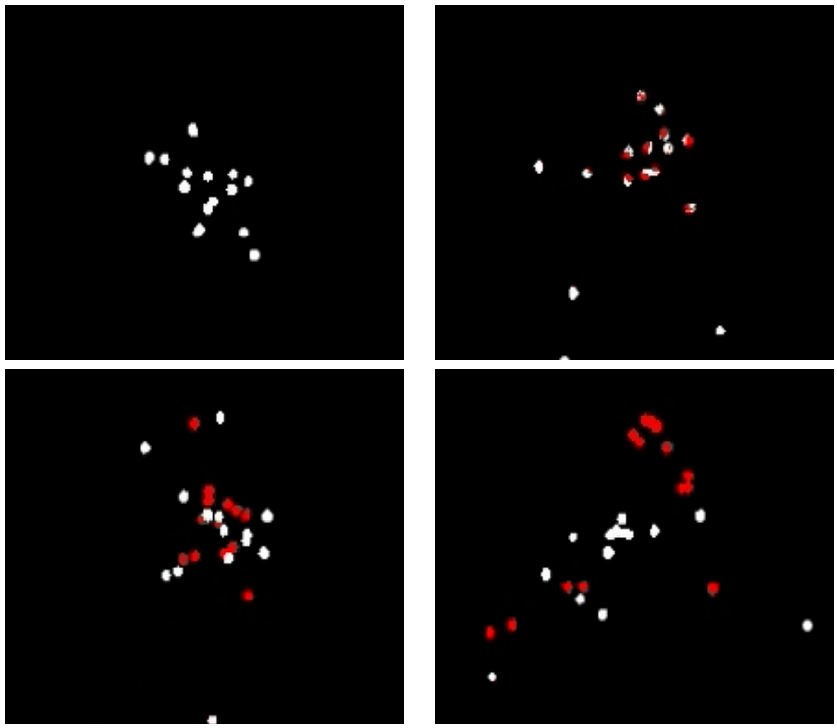

Figure 1.2: N-body simulation of a 16-body star cluster. The white stars represent the solution obtained by the Leapfrog method, and the red painted stars are obtained using the Hermite method. We show the initial condition (top row, left), and three snapshots a few crossing times apart (from left to right, and top to bottom), illustrating the divergence between neighbouring solutions.

1.3.2 Accuracy of N-body Results

The aim of any N-body simulation is to calculate the unique, mathematical solution to the N-body problem, given a set of initial conditions. The exact solution cannot generally be obtained analytically and so we have to resort to approximate, numerical methods. How do we know that the calculated solution is accurate?

The best way to quantify the accuracy of an approximate solution is to compare it to the true solution, but since the true solution is not available we need other methods. The most often used method is to check for energy conservation. As can be derived from the equations of motion, the total energy of a dynamical system is a conserved quantity. Any simulated N-body solution should not make too large energy errors in order to remain trustworthy (**?**). On the other hand, there are an infinite number of configurations on the energy hyperspace in the

multi-dimensional phase space, so that while the total energy might be conserved exactly, the precise configuration could still be completely wrong.

Another type of test is for reversibility. We start integrating the initial configuration up to some time t, then we change the direction of time and integrate backwards to $t = 0$. Ideally, we should reproduce the initial conditions, but due to error growth in the mean time, this might not be the case. For example, in a star cluster simulation consisting of single stars, the cluster will evolve towards core collapse, where its core density becomes extremely high and where binaries are formed due to three-body interactions (**?**). If we then reverse the arrow of time, we would expect the opposite evolution, where the core density gradually decreases and binaries get ionized. This backward evolution is however very unstable, and small deviations from this track will cause the evolution to go forwards again towards higher core density and binary formation.

Convergence tests are often performed to show that the obtained N-body results are safe and sound. After all, by decreasing the time step systematically, the discretisation error will become smaller up to the point that the first few decimal places will start to converge. The required precision however, depends on the specific configuration. For very chaotic systems, the necessary time step could be so small, that round-off errors due to the many time steps and operations, will become dominant and prevent any convergence. Every result obtained from N-body simulations should be treated carefully to make sure that we do not get led astray.

1.4 CHAOTIC DYNAMICS

Central in this thesis is the growth of a small initial perturbation in a dynamical system. Consider a slight perturbation of only one coordinate of a single body in the system. If we then calculate the accelerations of all the bodies, they will be slightly perturbed as well due to the slightly perturbed distances to the perturbed body. The new positions and velocities at a time step Δt later, will inherit the perturbation as well. Subsequent integration steps will magnify the perturbation until it has become the size of the system.

This mechanism is investigated from two different perspectives. First of all, we take the small initial perturbation to be a numerical error, so that we are looking at the growth of numerical noise in N-body simulations. As described in the previous section, it is challenging to obtain true solutions to the N-body problem due to different sources

of error (see Fig. 1.2). The main reason that small integration errors cause problems is that many dynamical systems are inherently chaotic. Similar to the familiar butterfly effect, a small integration error can be magnified by orders magnitude within a finite time (**???**). This limits the predictability of the positions of the bodies after a certain amount of time. To some degree, high-precision N-body methods can fight the exponentially growing numerical errors.

In the second interpretation, the small initial perturbation is physical and introduced manually in the initial conditions for the N-body experiment. By measuring the growth of the perturbation we can determine the stability of a dynamical system. If two neighbouring solutions remain close on the time scale of interest, then small perturbations do not alter the evolution significantly and we say the system is stable to small deviations. In this case the perturbation usually grows linear in time. If, on the other hand, a perturbed solution is completely different after a relatively short time, then the system is categorized as unstable and chaotic, and the perturbations grow exponentially.

Increasing our understanding of the way perturbations grow in dynamical systems is crucial for designing improved N-body methods and for determining the stability of dynamical systems and their long term evolution. One illustrative example is the stability of planetary systems. Hundreds of exoplanets have been discovered orbiting other stars and in a wide variety of orbits[2]. A significant fraction of exoplanetary systems are much more compact than our solar system. By understanding the reason why some configurations are stable and others unstable, we can learn about the evolution of planetary systems and the likelihood to observe them in a certain state.

1.5 THIS THESIS

This thesis presents five studies on new N-body algorithms, the reliability of N-body results, and finally the origin of chaos in dynamical systems. Below we introduce these topics and related research questions in somewhat more detail.

Chapter 2

As mentioned in Sec. 1.3.2, the best way to quantify the accuracy of an N-body simulation is to compare it to the true solution. We design the new N-body code Brutus which solves the N-body problem to a pre-defined precision. This code adaptively reduces the time step size

[2]http://exoplanets.org/

to control the discretisation errors, and it uses an arbitrary-precision library to control the round-off errors. By making the time step smaller and by increasing the number of used digits, we obtain solutions with an increasing precision up to convergence. These converged N-body solutions approach the true solution up to the first specified number of decimal places. Using this code we are able to quantify the accuracy of approximate solutions and for the first time obtain true solutions to general N-body configurations.

Chapter 3

We explore a conceptually novel idea for an N-body algorithm. It is based on two main arguments: the two-body problem can be solved analytically and the superposition principle holds in the calculation of the accelerations. Therefore, are we able to solve every pair of two-body problem in the system and combine them to solve the overall N-body problem? We demonstrate that this is indeed the case and perform several validation and performance tests. This code turns out to be very efficient on parallel supercomputers.

Chapter 4

The assumption in the N-body community is that even though N-body solutions inherently possess some numerical noise, the results are still valid statistically. We test this assumption for three-body scattering systems as described briefly in Sec. 1.1. We compare statistics on triple interactions and dynamically formed binary stars, obtained from an ensemble of converged solutions (obtained by Brutus) to those obtained from an ensemble of approximate solutions (obtained by Hermite). We find that for sufficient energy conservation, three-body scattering statistics are preserved under divergence of individual solutions, which is good news for the N-body user.

Chapter 5

In this Letter, we discuss in more detail the observation from the previous chapter, that an ensemble of approximate solutions preserves the global statistical distributions. We analyse the results from three-body scattering experiments and illustrate this quasi-ergodic prop-

erty of gravity, i.e. "Nagh-Hoch"[3], which exposes itself particularly in chaotic systems. This property is crucial for the reliability of N-body simulations, which inherently possesses numerical noise.

Chapter 6

Having determined in previous chapters that chaotic systems show an exponential growth of small perturbations, we now want to understand why these systems show this behaviour. Why are some systems chaotic and others regular and is it possible to go from order to chaos and vice versa? We construct a model for the growth of perturbations based on two-body Keplerian systems being perturbed by a third body. We apply the model to Halley's orbit and find its Liapounov time, i.e. the e-folding time for exponential growth, to be about 300 years. This surprisingly short time scale follows naturally from the encounter density (Halley and Jupiter are close to a 3:19 mean motion resonance) and the strength of each close encounter.

Chapter 7

The field of N-body code development and simulations has matured since the first simulations performed in the 1960s. There are however still many areas to explore or to improve. We briefly describe several open problems, which should be solved within the next decade.

[3]Klingon for "Ensemble of Stone", referring to the preservation of ensemble statistics under numerical errors.