Astrophysical Exercises: Orbits with MOdified Newtonian Dynamics

Joachim Köppen Strasbourg 2010

1. Physical Background

The big astronomical objects, such as galaxy clusters, galaxies, and stellar clusters are composed of many smaller objects (galaxies, stars, gas clouds) which interact gravitationally. This determines the structure and the evolution of these objects. However, it has been found (in the 1970s) that the disks of spiral galaxies revolve around the galactic centres with velocities which are nearly constant with distance from the centre. Such a rotation curve cannot be explained by gravitational attraction from the visible mass (stars and gas) that is inside the orbit of that part of the disk. One commonly accepted explanation is to postulate the existence and presence of "Dark Matter", a form of matter that is neither visible in emission nor in absorption, and whose interaction with ordinary "baryonic" matter is only through gravitation.

However, despite an intense search for any yet undiscovered particle that could play the role of dark matter, nothing has been found so far, and apart from the astronomical context of rotation of galaxies and similar dynamic discrepancies, no other field of physics has given evidence for the existence of such a kind of matter. Therefore, alternative explanations have been sought. One possibility could be that the gravitational law as formulated by Newton on the basis of the orbits of the planets of the Solar System, not longer applies or is somewhat altered for the large distances involved in galaxies. Another possibility is that Newton's second law of motion (that a body experiences an acceleration when a force is applied which is equal to the inertial mass times the acceleration) is no longer valid under these small values of acceleration we encounter in galaxies. In 1983, M.Milgrom proposed the idea of MOND = MOdified Newtonian Dynamics, showing the introduction of another universal and fundamental constant (the threshold of acceleration a_0 below which Newtonian gravity law would be modified) would be able to account for the observed galactic rotation curves. This idea has been further developed and more rigorous formulations have been found, so that this theory has become a serious alterative explanation of the dark matter paradigm.

In this exercise, we shall write a program to compute planetary/stellar orbits with Newtonian gravity as well as with MOND, in order to see how the orbits change. We shall consider both the case of a small test particle in a given fixed gravitational potential, as well as the proper two-body-problem, in which two bodies interact gravitationally.

2. The One-Body Problem

Let us first consider the simplest case: A small body (such as a planet) is in the gravitational field of a much more massive body ($M \gg m$), so that we may consider this gravity field as given by a constant potential:

$$\Phi = -\frac{GM}{r^2} \tag{1}$$

and the force experienced by the test mass is the gradient of the potential

$$\mathbf{Force} = -\frac{GMm}{r^2}\frac{\mathbf{r}}{r} \tag{2}$$

Newton's second law then gives the acceleration $\mathbf{a}$:

$$m\mathbf{a} = \mathbf{Force} \tag{3}$$

which changes the velocity and the position of the particle:

$$\frac{d^2}{dt^2}\mathbf{r} = \frac{d}{dt}\mathbf{v} = \mathbf{a} \tag{4}$$

These are all the equations that rule the movements of the test particle, and solving them permits us to compute its trajectory.

2.1 About units

For numerical computations it is always best, if the numbers the computer has to deal with are of the order of unity. Instead of using the real units of the problem - what would those be? In the Solar System cm, m, or even km give large numbers, and if we deal with galaxies, the distances would be even larger - it is best to work with scaled units. For instance, we see from eqns.(2) and (3) that the acceleration is proportional to a constant GM. If we choose units in which $GM = 1$ and place the test body at distance $r = 1$, we could get all results of the orbit and its shape. To interpret the results, we simply have scale the results to the proper dimensions: For instance, if we deal with the Earth's orbit around the Sun, $r = 1$ would mean that our unit of length is one astronomical unit ... We could amuse ourselves now to work out what should then be the unit of time, given that we might choose one solar mass as mass unit ... but let us see later!

2.2 How to Compute it

The equations (2) to (4) are vector equations, so we write them down for each component separately. For simplicity, we use cartesian coordinates $\mathbf{r} = (x, y, z)$. Here we can also introduce another simplification: Because of conservation of angular momentum, the orbit will always lie in a plane. Therefore, **it suffices to solve the equations in 2 dimensions**, which is less time consuming than working in the full three dimensions! For the x-component of the acceleration we will have

$$a_x = -\frac{GM}{r^2}\frac{x}{r} \tag{5}$$

To do the integration of time, we take a constant time step Δt, during which we assume that the acceleration at the 'old' time t is constant in each component. Thus the x-component of the velocity of the particle will change during this time step:

$$v_x(t + \Delta t) = v_x(t) + a_x \Delta t \tag{6}$$

Of course, during this time interval, the acceleration changes, as the mass point flies to another position. So what one really needs for the acceleration is a suitable mean value for each time step. But in order to compute this, we need to know the new position, ... which is what we are just going to compute. So, such a more accurate method (which is called *implicit* since it uses information about the new position) needs an iteration for each time step and it is more complex to program. For the time being, we shall use the simpler method, but please remember, it may be less accurate. It is called an *explicit* method, as it uses only information from the old time.

To compute the new position of the mass point at the new time $t + \Delta t$ we assume that the velocity is constant. However, the same question as before is raised: which velocity shall we use? But now we have two: the old one $v(t)$, or the new one $v(t + \Delta t)$, and we could even take some average value. We recommend to use the *new* velocities. The advantage of doing this is the following: One can simply show that the change of the angular momentum during a time step will be exactly zero, if we use this mixed method (Please check this by calculating analytically $\Delta\mathbf{L} = \mathbf{r}(t + \Delta t) \,\mathtt{X}\, \mathbf{v}(t + \Delta t) - \mathbf{r}(t) \,\mathtt{X}\, \mathbf{v}(t)$ from Eqs. (6) and (7), and also by observing the components of $\mathbf{L}$ in the numerical calculations). So the new positions are

$$x(t + \Delta t) = x(t) + v_x(t + \Delta t)\Delta t \tag{7}$$

Since we now use some information from the new time, the method is implicit. These computations are simply repeated for the next time time step ... as long as we wish to follow the test particle.

2.3 Checks of the Method

Before one applies one's program to complex problems, one must check whether it really does what it was designed to do. The ideal way is to run it for simple problems that have an analytical solution. For example:

1. compute the orbit around the Sun of a planet which has circular velocity: put the Sun at the centre ($x = y = 0$ and $v_x = v_y = 0$) and the planet at ($x = r$, $y = 0$) with initial velocity ($v_x = 0$, $v_y = v_c$). v_c is the velocity of a circular orbit with radius r.
2. derive from the equation of motion the circular speed for a system with $GM = 1$ and at radius $r = 1$? Is it $v = 1$ in our non-dimensional units? What would then be the period of this circular orbit? With these information, you will be able to scale your results to any real physical situation!
3. Now do the simulation: The planet should not only make a perfect circle around the Sun, and come back to the initial position (within one orbital period – which you had computed), but also keep staying on the circle for as many cycles as possible. The accuracy depends on the time step you had chosen.
4. the equations also give you the dependence of the circular speed as a function of initial radius. Check these predictions.
5. check other initial velocities. They should give elliptic orbits, and if one exceeds the escape velocity, the planet should never come back!
6. use the energy equation to find the escape velocity. Does your simulation agree with it? What is the ratio of escape and circular speed, for the same initial radius?
7. compute the kinetic and potential energies of the whole system, as well as the modulus or the components of the angular momentum vector. Plot them as a function of time. Since we do not feed in any energy or angular momentum, they must stay constant. Do they ???? How well does your program conserve energy and momentum? Dependence on time step!

2.4 Other Applications

Once the program is working properly, you can also play with the form of the potential: What happens to the orbits if you change the exponent for the dependence on distance? Suppose it were not -2 as in Newtonian gravity, but some other value! Or if it were positive? How do slightly elliptical orbits behave as a function of this exponent?

2.5 MOND

In its simplest form, the gravitational acceleration a computed by MOND is related to the newtonian gravitational acceleration g_N by this formula

$$a_N = a * \mu(|a|/a_0) \tag{8}$$

where μ is an interpolating function

$$\mu(x) = \begin{cases} 1 & \text{for } x \gg 1 \\ x & \text{for } x < 1 \end{cases} \tag{9}$$

and a_0 a value for the threshold acceleration, below which MOND modifies the gravitational acceleration. A simple continuously differentiable interpolating function is

$$\mu(x) = \frac{x}{1+x} \tag{10}$$

With this choice, you can solve eqn.(8) for a, the desired acceleration! (you'll get a quadratic equation, of which only one solution is physically sensible ... as $a_0 \to 0$ should converge against the newtonian case!)

From this we get the result that for accelerations much larger than a_0 we recover the standard Newtonian acceleration $a = a_N$, but for acceleration much smaller than a_0 (the "deep MOND regime") we get $a = \sqrt{a_0 * a_N}$.

This means that $|a| = GM/r^2$ changes to $|a| = \sqrt{a_0 GM}/r$ in the deep MOND regime, so that the gravitational attraction drops more slowly with distance ... this is the feature that permits MOND to explain the flat rotation curves of galactic disks in an easy way!

In our scaled system of units, the value of a_0 determines in which regime a planet would move, if we launch it at radius $r = 1$ and an initial speed v:

1. $a_0 = 0$ is the purely Newtonian situation
2. $a_0 \gg 1$ is the purely MOND situation
3. $a_0 = 1$ is a situation in which the planet is in the intermediate regime, being closer to Newton when accelerations are high (where in the orbit does this occur?) but closer to MOND when the acceleration is low. For our exercise, taking $a_0 = 1$ is a good choice to show basic effects
4. In real units one finds that about $a_0 = 10^{-9}$ m/s^2 reproduces the galactic rotation curves.

Now run simulations of situations similar to the ones done earlier with Newtonian gravity.

1. What is different?
2. Show that these differences are genuine, and not produced by insufficient numerical accuracy.

3. You will notice that the circular speed, computed with Newtonian gravity, does not result in a circular orbit. Change the initial speed until you get a circular orbit.
4. For a circular orbit the (modified) gravitational acceleration is equal to the centrifugal acceleration. Use this condition to compute the circular speed for any given value of a_0. Compare with the value you obtained from the simulations.
5. What about the escape speed?
6. In the very deep MOND limit $a_0 \gg 1$ the planet cannot escape from the potential. Why?

3. TwoBody Problem

Let us now consider the more general case of two bodies with arbitrary masses M_1, M_2 moving around each other. The fundamental equations are the same, here the one written for mass 1:

$$\frac{d^2}{dt^2}\mathbf{r}_1 = \frac{d}{dt}\mathbf{v}_1 = -\frac{GM_2}{|\mathbf{r}_1 - \mathbf{r}_2|^2}\frac{\mathbf{r}_1 - \mathbf{r}_2}{|\mathbf{r}_1 - \mathbf{r}_2|} \tag{11}$$

and the equivalent one for the second mass. If the initial positions and speeds were completely arbitrary, there would be apart from the motions of the bodies relative to each other also the linear motion at constant velocity of the centre-of-mass, whose position is given by:

$$\mathbf{r}_{\mathrm{COM}} = \frac{M_1\mathbf{r}_1 + M_2\mathbf{r}_2}{M_1 + M_2} \tag{12}$$

Its velocity is

$$\mathbf{v}_{\mathrm{COM}} = \frac{M_1\mathbf{v}_1 + M_2\mathbf{v}_2}{M_1 + M_2} \tag{13}$$

To avoid the uninteresting motion of the centre-of-mass, we may introduce relative coordinates $\mathbf{r}' = \mathbf{r} - \mathbf{r}_{\mathrm{COM}}$ and velocities $\mathbf{v}' = \mathbf{v} - \mathbf{v}_{\mathrm{COM}}$.

The other possibility is simply to prescribe initial conditions so that the centre-of-mass is at rest in our system of coordinates. For this purpose, let us define the mass ratio $\xi = M_2/M_1$ so that the two masses can be expressed as fractions of the total mass $M = M_1 + M_2$:

$$M_1/M = m_1 = \frac{1}{1+\xi}$$

$$M_2/M = m_2 = \frac{\xi}{1+\xi}$$

If we specify the initial **separation** of the two bodies by $\mathbf{d}_0$ and the **relative velocity** $\mathbf{v}_0$, the following recipe places the two bodies about the centre-of-mass at rest in the origin:

$$\mathbf{r}_1 = m_2\mathbf{d}_0 \qquad \mathbf{v}_1 = m_2\mathbf{v}_0$$

$$\mathbf{r}_2 = -m_1\mathbf{d}_0 \qquad \mathbf{v}_2 = -m_1\mathbf{v}_0$$

This choice is not unique: We could also have demanded to specify the initial position $\mathbf{r}_0$ of the second body relative to the centre-of-mass, as well as the relative velocity $\mathbf{w}_0$:

$$\mathbf{r}_2 = \mathbf{r}_0 \qquad \mathbf{v}_2 = \mathbf{w}_0$$
$$\mathbf{r}_1 = -\xi\mathbf{r}_0 \qquad \mathbf{v}_1 = -\xi\mathbf{w}_0$$

The equation of motion (eqn(11)) for the first body is

$$\frac{d}{dt}\mathbf{v}_1 = -\frac{GMm_2}{|\mathbf{r}_1 - \mathbf{r}_2|^2}\frac{\mathbf{r}_1 - \mathbf{r}_2}{|\mathbf{r}_1 - \mathbf{r}_2|}$$

We now use our scaled units and we set $GM = 1$ - but note that now M is the total mass of the system. Then we proceed as before.

Performing the simulations, we now also change the mass ratio ξ. If you have chosen the first setup of initial conditions, you will find that the circular speed - that is: the initial relative speed of the two bodies that results in circular orbits of both bodies about the centre-of-mass - is independent of the mass ratio. You can derive this from the equations!

In a way, these simulations yield the same kind of orbits as we had obtained before with Newtonian gravity. The only thing is that now both bodies orbit the centre-of-mass, and shapes of the two orbits are the same, only their major axes differ in accordance with the mass ratio!

4. TwoBody Problem in MOND

Let us now apply the modification of eqn(8) to the two-body problem! Note that you have to apply the MOND formula to either of the two equations of motion.

What do you get? It looks weird? Unhappy? Puzzled?

Check whether the results are caused by insufficient numerical accuracy! Check whether the basic laws of conservation of physics are obeyed ...

5. Again: TwoBody Problem in MOND

You may have convinced yourself that the odd behaviour is not caused by any bug in your programming and that it is not caused by the particular numerical method which we use. It is indeed something more fundamental: the bug is that the simple application of the MOND recipe is physically inconsistent. In particular, it violates Newton's third law ("actio = reactio"): the force with which the first body pulls on the second one is unequal to the force that the second body exerts on the first one!

This problem had been quickly realized after Milgrom first publication, and soon after a physically consistent formulation was developed. However, this is no longer a simple formula, because it involves a complete computation of a modified form of the gravitational potential.

For the two-body problem, very recently H.S.Zhao et al. have given a formula for the mutual force:

$$F_{12} = M_1 a_1 = M_2 a_2 = \frac{GM_1M_2}{r_{12}} + \frac{2}{3}\frac{\sqrt{Ga_0}}{r_{12}}((M_1 + M_2)^{3/2} - M_1^{3/2} - M_2^{3/2})$$

The first term on the right hand side is the Newtonian gravity law, the rest is the correction for MOND. Pure Newtonian gravity is recovered in the limit $a_0 \rightarrow 0$. Converting into our scaled units with $G(M_1+M_2) = 1$ and using the fractionalmasses m_1, m_2 we obtain

$$m_1 a_1 = m_2 a_2 = \frac{m_1 m_2}{r_{12}} + \frac{2}{3}\frac{\sqrt{a_0}}{r_{12}}(1 - m_1^{3/2} - m_2^{3/2}) \qquad (14)$$

Let us use this formulation to compute the accelerations, and let us see what we get!

1. In the limits of small or large mass ratios $\xi \ll 1$ and $\xi \gg 1$ we should get the same results we had obtained with the one-body simulation.
2. What is the circular speed? From eqn.14 you can work out the value expected for any initial separation r_{12}, mass ratio ξ, and a_0. Note that for our specification of the initial condition we had used v_0, the relative speed between the two bodies...
3. What about the escape speed?

6. General Remarks, Hints, and Kinks (I know you won't read this!!!)

0. Before actually writing the program, do make a flow chart diagram in order to understand the sequence of what is computed; a diagram of the program structure and the data structure to find out, how the loops and iterations are nested, which data from earlier parts you need at each section, which kind of vectors and arrays you are going to need. This may seem bureaucratic, boring or even old fashioned, but **don't start typing anything, before you are absolutely clear about what you plan to do**. Otherwise you may really end up wasting much time in trying to find the logical errors, loopholes, and cul-de-sacs of your hasty programming. Save yourself the frustration, disappointment, and anger!
1. General Program Planning: It is a good idea to lay out the program as general as possible. This makes it easier to include other effects, or to try out other situations. Expanding the program to other force laws, e.g. repelling, as for the a cluster of protons instead of stars, or completely different dependence on distance.
2. Modular Construction: It is also a good idea to break up the program into mathematically or physically sensible units. This allows a better testing of these

individual modules — and most of the time is spent in tracing an error — a more flexible use of them for other purposes, and their exchange against improved or alternative methods, improved data, other physical processes, etc. For example, if the time integration is contained in a separate unit, one simply exchanges this against a more sophisticated method, if need arises, but without the trouble of having to change the program at a dozen places. For testing, a simple main program has to be written which supplies the necessary input data to this unit. When adapting the program to a different problem, one merely re-arranges the modules. The main program may then be just a control program to call the subprograms in the particular order, and gets and supplies data from and to each part.

3. Check Everything by Hand: Often, we underestimate our ingenuity to make small logical mistakes or simple typing errors, which may cause faulty results. The worst kind of mistakes are those which produce results that look as one would expect them to be. Do take the trouble of check everything the program does, until you are sure it does only what you want it to do. In programs about physical things, basic physics must be obeyed: conservation of particles, energy, etc. Also, all the simple and limiting cases which we do understand, must be reproduced accurately.
4. Provide Error Messages and Tracing: Especially during testing, you will print out everything that is useful to judge on what the program does and what decisions it makes. For example, have a print out of the energies and angular momentum of the system. It is recommendable to define one or more variables that can be used to switch on these print-outs. In this way, one can always check every part of the program, even after it has been considered finished. If you later want to try out some modification, and the results are either complete rubbish (because you made some error) or you just want to understand how the solution behaves, this option is very useful. Provide written messages if something goes not normal, e.g. if the automatic stepwidth goes below or above specified values. This become more important as the program grows in size and complexity.
5. Take Time For Comments: Don't be lazy with putting comments into the program. Not only for the general description what each subprogram does, what data it needs, and what variables it changes. But also if you change just a line for a test. Often one forgets after a few days about it, and is quite surprised about the results. Save yourself the panic! Plenty of comments are vital, if you find some time later that you might use it for something, but you can't remember about its inner workings. Don't wait for them after "the program is finished". It never happens, or you won't have the time.
6. Be highly skeptic of anything that the program produces.
7. When you are making tests, and later running the program for various situations and parameters, try to keep a careful written record of what you do, noting input parameters and results. This will make it easier for you later to compare results with earlier ones, in case you have to hunt for an error that has crept in yesterday

when you “just changed a few things, almost nothing — but the program doesn’t work any more”.

7. Simulations for Comparison

At `http://astro.u-strasbg.fr/~koppen/MOND/` you will find Java applets for the one-body and two-body problems, for Newtonian and MOND formulations, and with a variety of numerical methods.