UNIVERSITAT POLITÈCNICA DE CATALUNYA

ENGINYERIA FÍSICA

ASTROPHYSICS and COSMOLOGY

Project Work 3

FREE-FALL COLLAPSE OF A HOMOGENEOUS SPHERE

Alexandre Justo and José Javier Ruiz

24/12/2018

Abstract

Two main forces are present in stars: the gravity force, which tends to shrink the star; and the interior pressure force, which sustains the star and prevents it from collapsing. This pressure may come from nuclear reactions, which take place in the interior of those stars that have not consumed its hydrogen fuel, or from the electron or neutron degeneracy, if it refers to white dwarfs or neutron stars, respectively.

This article makes a numerical approach to the free-fall collapse of a star, considering it as a sphere discretized in N shells with the same mass each. Dynamical equations are obtained through mass conservation equation, momentum conservation equation and Lagrangian velocity equation, by considering there is no pressure at all within the star.

Contents

1 Introduction

In this project a suitable computational grid has been implemented in the sphere, by dividing it into N concentric shells with the same mass. This leads to a discretization of the mass conservation, momentum conservation and Lagrangian velocity expressions, by changing the derivatives by their finite difference forms. Is in these discretizations where the integration parameter β takes a primordial role.

In order to solve the dynamical behaviour of each shell, a linearization has been done, where the objective is to compute the value of the smalls corrections for each instant of time. To compute them, it has been taken advantage of the fact that the equations for one shell are uncoupled from the other shells. Then, instead of working with a matrix of dimensions $3N \times 3N$, it has been possible to work with N matrices of dimensions 3×3. In addition, the analytic solution (in continuous form) of the evolution of the star radius $R(t)$ has been obtained, which has been used to compare the accuracy of the numerical results.

2 Numerical approach

In order to simulate a free-fall collapse, three main equations are to be considered.

The mass conservation equation is given by:

$$\frac{1}{\rho} = \frac{4}{3}\pi \frac{\partial r^3}{\partial m} \tag{1}$$

After considering no pressure and no artificial viscosity in the sphere, the momentum conservation equation can be written as follows:

$$\frac{\partial u}{\partial t} = -G\frac{m}{r^2} \tag{2}$$

The other equation to be considered is the one regarding Lagrangian velocity:

$$\frac{\partial r}{\partial t} = u \tag{3}$$

By discretizing those three, one can have a starting point for building a numerical model. One can divide the sphere into N concentric shells.

Discretized version of conservation of mass:

$$\frac{1}{\rho_{i+1/2}^{n+1}} = \frac{4}{3}\pi \frac{(r_{i+1}^{n+1})^3 - (r_i^{n+1})^3}{m_{i+1} - m_i} \tag{4}$$

Discretized version of conservation of momentum:

$$\frac{u_{i+1}^{n+1} - u_{i+1}^{n}}{\Delta t} = (1-\beta)\left(\frac{-Gm_{i+1}}{r_{i+1}^2}\right)^n + \beta\left(\frac{-Gm_{i+1}}{r_{i+1}^2}\right)^{n+1} \tag{5}$$

Discretized version of Lagrangian velocity:

$$\frac{r_{i+1}^{n+1} - r_{i+1}^{n}}{\Delta t} = (1-\beta)u_{i+1}^{n} + \beta u_{i+1}^{n+1} \tag{6}$$

Where the subindexes denote the variables being evaluated at the i^{th} shell, with $i = 1, ..., N$ and the superindexes denote the variables being evaluated at the n^{th} instant of time. For the case of the density ρ, it will be evaluated at the midpoints between shells, unlike all the other variables, which will be evaluated at the shells.

The constant β is an integration parameter ($0 \leq \beta \leq 1$) that defines different numerical schemes. The choice $\beta = 0$ defines the explicit methods, whereas the choice $\beta \neq 0$ defines implicit methods. Several different values of β will be imposed to obtain the numerical solution of the free-fall collapse, which will be compared with the analytical solution in order to study which value gives the most accurate result.

This model defines the shells in such a way that each shell contains the same amount of mass during all the simulation. That is

$$m_{i+1} = i\Delta m = i\frac{M}{N} = i\frac{\frac{4}{3}\pi R_0^3 \rho_0}{N} \tag{7}$$

Where R_0 is the initial radius of the star and ρ_0 is the initial density of the star. Those two values will be inputs of the code, and in this article $R_0 = 10^4\ cm$ and $\rho_0 = 10^7 g\ cm^{-3}$ are chosen.

As boundary conditions, one must choose the following

$$r_1^n = 0 \tag{8}$$

$$m_1^n = 0 \tag{9}$$

Since the model considers the entire sphere, from its center to its surface, therefore the first shell must model the center. These conditions must be fulfilled for all time steps n.

As initial conditions, by definition of a homogeneous sphere, $\rho_{i+1/2}^0 \equiv \rho_0$ $(i = 1, N)$, and assuming an initial static configuration, $u_i^0 = 0$ $(i = 1, N+1)$ at $t = 0$.

By applying the conservation of mass (4) it is possible to obtain the initial radii of the shells. Firstly, for the innermost shell, considering that $m_1 = r_1 = 0$:

$$r_2^0 = \left(\frac{3m_2}{4\pi\rho_{3/2}^0}\right)^{1/3} \tag{10}$$

Similarly, the initial radii of the subsequent shells ($i = 2, N$) can be obtained through the following expression:

$$r_{i+1}^0 = \left(\frac{3(m_{i+1} - m_i)}{4\pi\rho_{i+1/2}^0} + (r_i^0)^3 \right)^{1/3} = \left(\frac{3\Delta m}{4\pi\rho_{i+1/2}^0} + (r_i^0)^3 \right)^{1/3} \tag{11}$$

In order to perform a future linearization of the dynamical equations, it is useful to express the discretized version of conservation of mass (4), conservation of momentum (5), and of Lagrangian velocity (6) as a function of the parameters, being all those expressions equal to zero. To do that, it is necessary to differentiate between the innermost shell and intermediate and outermost shells. Let us start first with the innermost shell.

Firstly, the mass conservation (10):

$$C^1(r_2, \rho_{3/2}) = \frac{1}{\rho_{3/2}^{n+1}} - \frac{4}{3}\pi \frac{(r_2^{n+1})^3}{m_2} = 0 \tag{12}$$

Secondly, the conservation of momentum (5):

$$C^2(u_2, r_2) = \frac{u_2^{n+1} - u_2^n}{\Delta t} - (1-\beta)\frac{-Gm_2}{(r_2^n)^2} - \beta\frac{-Gm_2}{(r_2^{n+1})^2} = 0 \tag{13}$$

And finally, the Lagrangian velocity (6):

$$C^3(u_2, r_2) = \frac{r_2^{n+1} - r_2^n}{\Delta t} - (1-\beta)u_2^n - \beta u_2^{n+1} = 0 \tag{14}$$

In general, this set of equations can be written as a function of just 3 unknowns, u_2, r_2, and $\rho_{3/2}$, such that

$$C^j(\rho_{3/2}, u_2, r_2) = 0 \ (j = 1, 3) \tag{15}$$

The following step is doing the same with the $N-2$ intermediate shells and the outermost shell ($i = 2, N$):

$$F_i^1(r_{i+1}, r_i, \rho_{i+1/2}) = \frac{1}{\rho_{i+1/2}^{n+1}} - \frac{4}{3}\pi \frac{(r_{i+1}^{n+1})^3 - (r_i^{n+1})^3}{m_{i+1} - m_i} = 0 \tag{16}$$

$$F_i^2(u_{i+1}, r_{i+1}) = \frac{u_{i+1}^{n+1} - u_{i+1}^n}{\Delta t} - (1-\beta)\frac{-Gm_{i+1}}{(r_{i+1}^n)^2} - \beta\frac{-Gm_{i+1}}{(r_{i+1}^{n+1})^2} \tag{17}$$

$$F_i^3(u_{i+1}, r_{i+1}) = \frac{r_{i+1}^{n+1} - r_{i+1}^n}{\Delta t} - (1-\beta)u_{i+1}^n - \beta u_{i+1}^{n+1} = 0 \tag{18}$$

As with the innermost shell, this set of equations can be written as a function of now 4 unknowns.

$$F_i^j(\rho_{i+1/2}, r_i, r_{i+1}, u_{i+1}) = 0 \ (i = 2, N; \ j = 1, 3) \tag{19}$$

The strategy followed for implementing a numerical code with the purpose of solving the free-fall problem is being explained. Small increases for each of the variables will be calculated. Those variables are $\rho_{i+1/2}$, r_i, r_{i+1}, u_{i+1}. This

way, those variables calculated at the time step $n+1$ will be the ones calculated at the time step n plus the aforementioned increase. For instance, for the case of the density

$$\rho_{i+1/2}^{n+1} = \rho_{i+1/2}^{n} + \delta\rho_{i+1/2}^{n} \tag{20}$$

How to obtain those δ increases? For small corrections, the whole set of structure equations can be written in a form that corresponds to the following system of linearized equations [2]:

$$C^j + \frac{\partial C^j}{\partial \rho_{3/2}}\delta\rho_{3/2} + \frac{\partial C^j}{\partial r_2}\delta r_2 + \frac{\partial C^j}{\partial u_2}\delta u_2 = 0 \tag{21}$$

$$F_i^j + \frac{\partial F_i^j}{\partial \rho_{i+1/2}}\delta\rho_{i+1/2} + \frac{\partial F_i^j}{\partial r_i}\delta r_i + \frac{\partial F_i^j}{\partial r_{i+1}}\delta r_{i+1} + \frac{\partial F_i^j}{\partial u_{i+1}}\delta u_{i+1} = 0 \tag{22}$$

There appear several partial derivatives inside expressions (21) and (22). It is convenient to write them explicitly, since they must be computed.

Here, note that two different time steps are within the equations. The last calculated and converged time step will be treated as a constant when performing the derivatives, since it will not change anymore, and the only variables to change are those corresponding to the time step $n+1$.

The ones derived with respect to $\rho_{i+1/2}$:

$$\frac{\partial F_i^1}{\partial \rho_{i+1/2}} = -\frac{1}{(\rho_{i+1/2})^2} \tag{23}$$

$$\frac{\partial F_i^2}{\partial \rho_{i+1/2}} = 0 \tag{24}$$

$$\frac{\partial F_i^3}{\partial \rho_{i+1/2}} = 0 \tag{25}$$

The ones derived with respect to r_i:

$$\frac{\partial F_i^1}{\partial r_i} = 4\pi\frac{(r_i)^2}{\Delta m} \tag{26}$$

$$\frac{\partial F_i^2}{\partial r_i} = 0 \tag{27}$$

$$\frac{\partial F_i^3}{\partial r_i} = 0 \tag{28}$$

The ones derived with respect to r_{i+1}:

$$\frac{\partial F_i^1}{\partial r_{i+1}} = -4\pi\frac{(r_{i+1})^2}{\Delta m} \tag{29}$$

$$\frac{\partial F_i^2}{\partial r_{i+1}} = -2\beta\frac{G\ m_{i+1}}{(r_{i+1})^3} \tag{30}$$

$$\frac{\partial F_i^3}{\partial r_{i+1}} = \frac{1}{\Delta t} \tag{31}$$

The ones derived with respect to u_{i+1}:

$$\frac{\partial F_i^1}{\partial u_{i+1}} = 0 \tag{32}$$

$$\frac{\partial F_i^2}{\partial u_{i+1}} = \frac{1}{\Delta t} \tag{33}$$

$$\frac{\partial F_i^3}{\partial u_{i+1}} = -\beta \tag{34}$$

The same must be done for the innermost shell equations, C^j, yielding very similar results.

Thus, the objective now is to determine the three corrections $\delta\rho^n_{i+1/2}, \delta r^n_{i+1}, \delta u^n_{i+1}$ for each shell and time step. This can be done in a matrix way; however, notice that there are three corrections for N shells, then, it would consist on solving a system with a matrix dimension of $3N \times 3N$ for each time step. Considering that N has been chosen to be $N = 100$, solving this system is not easy to handle numerically. Fortunately, the system of equations for each shell can be treated as uncoupled from the other shells, which allows to work with N matrices of dimension 3×3, instead of just one matrix of $3N \times 3N$.

Let us start with the matrix form of the system of equations corresponding to the innermost shell (21):

$$\begin{pmatrix} \frac{\partial C^1}{\partial \rho_{3/2}} & \frac{\partial C^1}{\partial u_2} & \frac{\partial C^1}{\partial r_2} \\ \frac{\partial C^2}{\partial \rho_{3/2}} & \frac{\partial C^2}{\partial u_2} & \frac{\partial C^2}{\partial r_2} \\ \frac{\partial C^3}{\partial \rho_{3/2}} & \frac{\partial C^3}{\partial u_2} & \frac{\partial C^3}{\partial r_2} \end{pmatrix} \begin{pmatrix} \delta\rho_{3/2} \\ \delta u_2 \\ \delta r_2 \end{pmatrix} = \begin{pmatrix} -C^1 \\ -C^2 \\ -C^3 \end{pmatrix} \tag{35}$$

Writing the partial derivatives explicitly:

$$\begin{pmatrix} -\frac{1}{(\rho_{3/2})^2} & 0 & -4\pi\frac{(r_2)^2}{m_2} \\ 0 & \frac{1}{\Delta t} & -\frac{\beta G m_2}{(r_2)^2} \\ 0 & -\beta & \frac{1}{\Delta t} \end{pmatrix} \begin{pmatrix} \delta\rho_{3/2} \\ \delta u_2 \\ \delta r_2 \end{pmatrix} = \begin{pmatrix} -C^1 \\ -C^2 \\ -C^3 \end{pmatrix} \tag{36}$$

Solving this system, the corrections $\delta\rho_{3/2}$, δu_2, δr_2 can be obtained. Later, the same has to be done with the intermediate and outermost shells, one after another. Let us compute the matrix form of (22):

$$\begin{pmatrix} \frac{\partial F_i^1}{\partial \rho_{i+1/2}} & \frac{\partial F_i^1}{\partial u_{i+1}} & \frac{\partial F_i^1}{\partial r_{i+1}} \\ \frac{\partial F_i^2}{\partial \rho_{i+1/2}} & \frac{\partial F_i^2}{\partial u_{i+1}} & \frac{\partial F_i^2}{\partial r_{i+1}} \\ \frac{\partial F_i^3}{\partial \rho_{i+1/2}} & \frac{\partial F_i^3}{\partial u_{i+1}} & \frac{\partial F_i^3}{\partial r_{i+1}} \end{pmatrix} \begin{pmatrix} \delta\rho_{i+1/2} \\ \delta u_{i+1} \\ \delta r_{i+1} \end{pmatrix} = \begin{pmatrix} -F_i^1 - \frac{\partial F_i^1}{\partial r_i}\delta r_i \\ -F_i^2 - \frac{\partial F_i^2}{\partial r_i}\delta r_i \\ -F_i^3 - \frac{\partial F_i^3}{\partial r_i}\delta r_i \end{pmatrix} \tag{37}$$

Notice that δr_i can be at the right hand-side of the equations, since it has been already computed in the previous shell system. Writing the partial derivatives explicitly:

$$\begin{pmatrix} -\frac{1}{(\rho_{i+1/2})^2} & 0 & -4\pi\frac{(r_{i+1})^2}{\Delta m} \\ 0 & \frac{1}{\Delta t} & -\frac{2\beta G m_{i+1}}{(r_{i+1})^3} \\ 0 & -\beta & \frac{1}{\Delta t} \end{pmatrix} \begin{pmatrix} \delta\rho_{i+1/2} \\ \delta u_{i+1} \\ \delta r_{i+1} \end{pmatrix} = \begin{pmatrix} -F_i^1 - \frac{4\pi(r_i)^2}{\Delta m}\delta r_i \\ -F_i^2 \\ -F_i^3 \end{pmatrix} \quad (38)$$

Finally, to obtain the values of the dynamical variables at time step $n+1$, the corrections δ obtained must be added to the value of the dynamical variables at time step n. This process must be done iteratively until arriving to the desired final time of integration.

However, before advancing one time step, one must ensure that the δ corrections are accurate enough. For that purpose, the process of calculating those corrections for all shells is going to be improved, for the same time step, until $\delta x/x < \epsilon$, where here the tolerance ϵ has been chosen to be 10^{-6}. The way of improving them is by firstly calculating the small corrections and adding them to the actual variables. With these updated variables, and without advancing a time step yet, the same process is repeated, and the new corrections to the updated values are checked. If only one of them, taking all shells into account, does not satisfy $\delta x/x < \epsilon$, the corrections are added again and the variables are updated now for the second time, and with these new variables the same process is repeated, corrections are checked until the point where absolutely all corrections for all shells satisfy $\delta x/x < \epsilon$. Then and only then, the updated variables are taken as definitive for that time step, and the time is advanced one step. This process will counteract the imprecision of artificially decoupling the big $3N \times 3N$ matrix, thus allowing the artificially decoupled variable δr_i not to suffer any errors.

Apart from that, it is worthy to discuss which time step and which final time of integration should be chosen. Actually, none of them are fixed a priori.

The initial time step is chosen to be $\Delta t = 0.01\ s$. However, it is not going to remain constant along all the simulation. When one runs the simulation with this fixed time step, they realise that the last steps of the simulation feature dramatic changes in the radius of the star. There are two main objectives in this project: to determine when the star reaches a radius equal to the 1% of the initial radius, and to determine when the numerical radius of the star differs in a pc % from the analytical radius of the star, where for instance $pc = 10$ means a difference of 10 %. In order to acquire good precision for that, the time step should be much lower than the initial one. However, the smaller the time step, the slower the simulation. A trade-off must be discussed here. In the end, the best way to deal with both problems is to make use of an adaptive time step.

So in the code created for this project, time steps are advanced until the radius of the star is below $0.01R_0$ if pc is chosen to be $pc = 0$, or until the radius of the star differs in a pc % from the analytical value at the same exact time if $0 < pc < 100$. When at some step the corresponding condition applies, it is not enough to stop the loop and store the last step variables, because then the precision with the time step of $\Delta t = 0.01\ s$ would be really poor. In order to gain precision, this code automatically invalidates the last iteration, reduces the

time step so that, with an abuse of notation, $\Delta t = \Delta t/2$, and calculates again the critical step. This is not repeated infinitely, because all this algorithm is located inside a loop that runs until the time step is not higher than 10^{-6} s. Therefore, the precision in time is guaranteed to be of at least 10^{-6} s, and the resulting values of the radius are going to be very close to the values desired, as it will be proved in the next sections.

3 Analytical solution

Start from equation (2). Rewrite the left hand side so that a change of variable is made. Use the chain rule:

$$\frac{\partial}{\partial t}u = \frac{\partial r}{\partial t}\frac{\partial}{\partial r}u \tag{39}$$

Plug equation (3) into equation (39) and write the missing right hand side of equation (2):

$$u\frac{\partial}{\partial r}u = -G\frac{M}{r^2} \tag{40}$$

$$\frac{1}{2}\frac{\partial}{\partial r}u^2 = -G\frac{M}{r^2} \tag{41}$$

$$d\left(u^2\right) = -2G\frac{M}{r^2}dr \tag{42}$$

Integrate it, by considering initial zero velocity and initial radius R_0, and obtain:

$$u^2 = 2GM\left(\frac{1}{R} - \frac{1}{R_0}\right) \tag{43}$$

Take the square root and use the Lagrangian velocity expression. Consider only the negative determination of the square root, since the radius is only expected to decrease:

$$\frac{dR}{dt} = -\sqrt{2GM\left(\frac{1}{R} - \frac{1}{R_0}\right)} \tag{44}$$

Prepare the expression to be integrated:

$$\frac{-dR}{\sqrt{2GM\left(\frac{1}{R} - \frac{1}{R_0}\right)}} = dt \tag{45}$$

Use a change of variable, $\alpha = \frac{R}{R_0}$, with $d\alpha = \frac{dR}{R_0}$:

$$dt = -\left(\frac{2GM}{R_0^3}\right)^{-1/2}\left(\frac{\alpha}{1-\alpha}\right)^{1/2}d\alpha \tag{46}$$

Use now $\frac{M}{R_0^3} = \frac{4}{3}\pi\rho_0$ and rewrite:

$$dt = -\left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}\left(\frac{\alpha}{1-\alpha}\right)^{1/2}d\alpha \tag{47}$$

This expression can be simplified with another change of variable, namely $\alpha = \sin^2\phi$, with $d\alpha = 2\sin\phi\cos\phi d\phi$, and by using $2\sin^2\phi = 1 - \cos(2\phi)$, yields:

$$dt = -\left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}(1-\cos(2\phi))d\phi \tag{48}$$

Now integrate from initial time t_0 to time t, considering that at initial time $R = R_0$.

$$(t-t_0) = -\left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}\int_1^{\arcsin\sqrt{\frac{R}{R_0}}}(1-\cos(2\phi))d\phi \tag{49}$$

$$(t-t_0) = -\left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}\left[\phi - \frac{1}{2}\sin(2\phi)\right]_{\phi=\arcsin 1}^{\phi=\arcsin\sqrt{\frac{R}{R_0}}} \tag{50}$$

Now evaluate the integration limits. Using the relations $\sin(\arcsin(x)) = x$, and $\sin(2x) = 2\cos(x)\sin(x) = 2\sin(x)\sqrt{1-\sin(x)^2}$, the result is:

$$(t-t_0) = \left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}\left[\phi - \sin(\phi)\sqrt{1-\sin(\phi)^2}\right]_{\phi=\arcsin\sqrt{\frac{R}{R_0}}}^{\phi=\arcsin 1} \tag{51}$$

$$(t-t_0) = \left(\frac{8\pi G\rho_0}{3}\right)^{-1/2}\left[\frac{\pi}{2} - \arcsin\left(\sqrt{\frac{R}{R_0}}\right) + \sqrt{\frac{R}{R_0}}\sqrt{1-\frac{R}{R_0}}\right] \tag{52}$$

From trigonometric relationships, it is known that $\pi/2 - \arcsin x = \arccos x$. Furthermore, assuming $x = \sqrt{R/R_0}$ is positive, then $\arccos x = \arcsin\sqrt{1-x^2}$. With all this, the final result is given by:

$$\left(\frac{8\pi G\rho_0}{3}\right)^{1/2}(t-t_0) = \sqrt{\frac{R}{R_o}}\sqrt{1-\frac{R}{R_o}} + \arcsin\left(\sqrt{1-\frac{R}{R_o}}\right) \tag{53}$$

That is, this is the analytic solution of a free-fall collapse of a homogeneous sphere. It will be used to compare the numerical results obtained and discussed in the following sections.

4 Computation and results

There follow all the graphs obtained. The interpretation of the results is discussed.

In all the simulations, the parameters of the star are: number of shells $N = 100$, initial radius $R_0 = 10^4 cm$ and initial density $\rho_0 = 10^7 g\ cm^{-3}$.

To begin with, the numerical result of the evolution of the star radius $R(t)$ with an integration parameter $\beta = 0.5$ is plotted, along with the analytic solution given by (53).

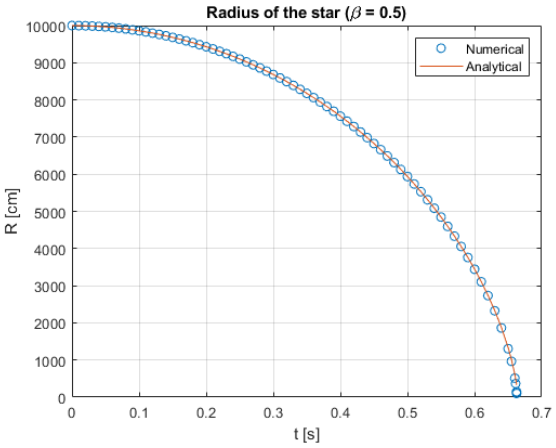

Figure 1 Free fall collapse of a homogeneous sphere, with an integration parameter $\beta = 0.5$.

As it can be inferred from Figure 1, analytical and numerical solutions match excellently, what guarantees that the implemented code works. Expectedly, the radius of the star decreases as time advances, furthermore, it decreases faster as time advances, until completely collapsing within a time lower than 0.7 s.

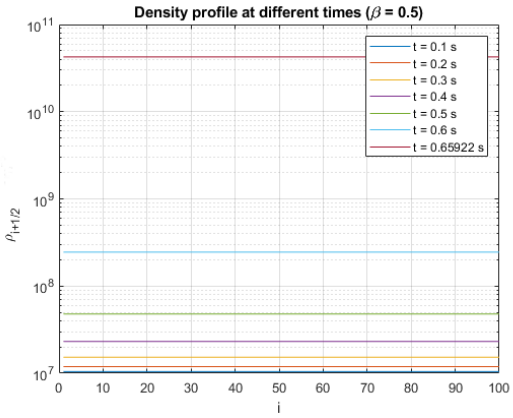

Figure 2 Density in units of $[g\ cm^{-3}]$ for each shell. The sphere remains homogeneous during all the process because all the curves are straight lines with zero slope.

For $\beta = 0.5$, the time required to shrink the radius of the sphere to 1% of its initial radius is 0.662680 s, calculated with a precision of 10^{-6} s.

In Figure 2 there have been represented the corresponding densities of each shell i for different instants of time. As it can be inferred from it, for each instant of time, the density is equal in all shells. That is, the sphere remains homogeneous during the collapse. Although they will not be shown in this ar-

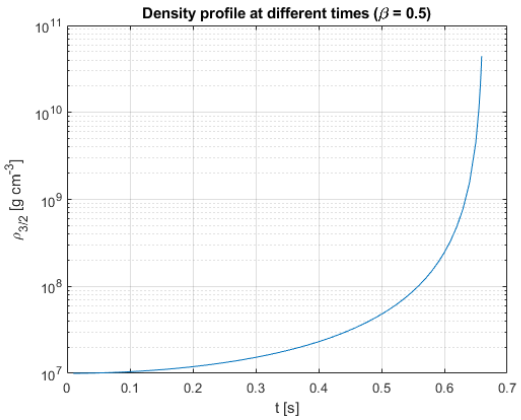

Figure 3 Central density as a function of time. The sphere homogeneously increases in density as time advances. For orders of magnitude of $10^{10} g\ cm^{-3}$, the sphere reaches a radius of $0.01 R_0$.

ticle for reasons of simplicity, no matter the value of the parameter β, all the density profiles correspond to a homogeneous sphere, thus yielding diagrams very similar to that of Figure 2.

In addition, as shown in Figure 3, the density of each shell increases with time. This behaviour was expected due to the fact that the total mass of the star is constant, but its total radius decreases during the collapse.

After discussing the results obtained for, according to theory, the best choice of the integration parameter, $\beta = 0.5$, now the results for other values of β are discussed. All those results will be gathered in Table 1.

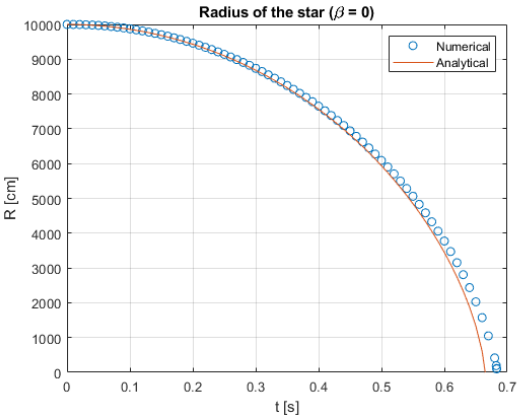

Figure 4 Free fall collapse of a homogeneous sphere, with an integration parameter $\beta = 0$.

For $\beta = 0$, the time required to shrink the radius of the sphere to 1% of its

initial radius is 0.683185 s, calculated with a precision of 10^{-6} s. See Figure 4.

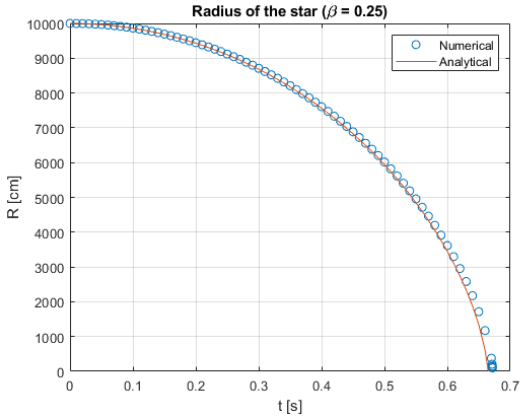

Figure 5 Free fall collapse of a homogeneous sphere, with an integration parameter $\beta = 0.25$.

For $\beta = 0.25$, the time required to shrink the radius of the sphere to 1% of its initial radius is 0.671807 s, calculated with a precision of 10^{-6} s. See Figure 5.

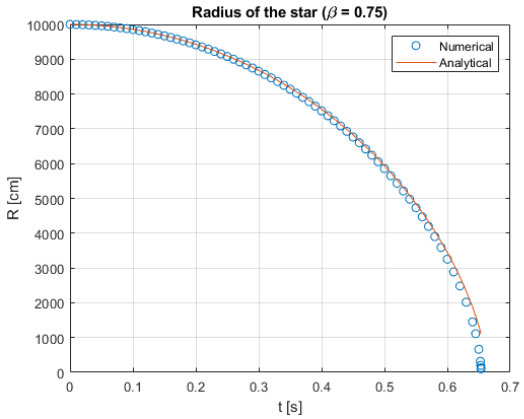

Figure 6 Free fall collapse of a homogeneous sphere, with an integration parameter $\beta = 0.75$.

For $\beta = 0.75$, the time required to shrink the radius of the sphere to 1% of its initial radius is 0.653561 s, calculated with a precision of 10^{-6} s. See Figure 6.

For $\beta = 1$, the time required to shrink the radius of the sphere to 1% of its initial radius is 0.645717 s, calculated with a precision of 10^{-6} s. See Figure 7.

It is possible to compute the theoretical time required to achieve just 1% of the initial radius R_0 through the analytic solution (53). That is, considering

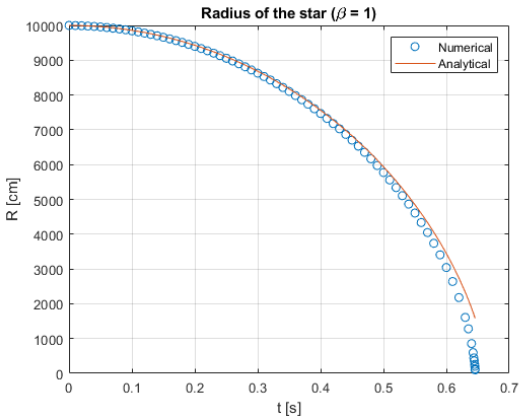

Figure 7 Free fall collapse of a homogeneous sphere, with an integration parameter $\beta = 1$.

$t_0 = 0$ and $R = 0.01R_0$, the theoretical time required is 0.664092 s. With the aim of determining which β is more accurate, in Table 1 are gathered the numerical times obtained for each β and their relative error with respect to the theoretical value (positive if it has been overestimated, negative if it has been underestimated).

Table 1 Time required for the sphere to shrink to 1% of its initial radius, with a precision of 10^{-6} s.

β	$t_{1\%}$	Relative error
0	0.683185 s	2.88 %
0.25	0.671807 s	1.16 %
0.50	0.662681 s	-0.21 %
0.75	0.653561 s	-1.57 %
1	0.645717 s	-2.77 %

Taking into account the relative errors shown in Table 1, it can be concluded that the more accurate interpolation parameter is $\beta = 0.5$, and the worst one is $\beta = 0$ (explicit method).

The next section is aimed to calculate the final value of R for which significant differences between the numerical and the analytical solution appear. The code is prepared such that, when a difference of 10% with respect to the analytical solution appears, the simulation stops and the results show up. This was repeated for the same values of β from the previous section.

The results are gathered together in Table 2.

Notice that the smaller the Numerical R, or larger time t, the more accurate is parameter β. Hence, taking into account these parameters, the best value of

Table 2 Final value of R for which a difference of 10 % appears between the numerical and the analytical solution.

β	Numerical R	Analytic R	t
0	3687.15 cm	3351.96 cm	0.602880 s
0.25	2625.63 cm	2386.94 cm	0.628781 s
0.5	608.30 cm	675.87 cm	0.659316 s
0.75	2399.59 cm	2666.20 cm	0.621913 s
1	3297.80 cm	3664.22 cm	0.593170 s

the interpolation parameter is $\beta = 0.5$, and the worst $\beta = 0$ (explicit method). Exactly, same results were obtained before.

5 Conclusions

Comparing numerical and analytic solution of $R(t)$, it can be concluded that the discretization of the sphere has been done properly, since both results match very well, with little differences depending on the interpolation parameter β. It has been shown that the best interpolation parameter is $\beta = 0.5$, and the worst $\beta = 0$, which corresponds to an explicit method. Another conclusion extracted from the results, is that the density of each shell is the same at every instant of time, then, a homogeneous sphere remains homogeneous while it collapses.

Finally, another important point that must be stressed, is the necessity of implementing an adaptive time step of integration in order to guarantee good precision. Otherwise, this free-fall collapse problem is very difficult to solve accurately in a reasonable computation time.

6 Source Code

There follows the MATLAB script used for the purpose of this project. It is commented step by step.

```
% General parameters.
N = 100; R0 = 1e4; rho0 = 1e7; dt = 0.01;

% Parameter percentage.
pc = 10;

% Select pc = 0 for the simulation to run until a 1% of the ...
    initial radius
% is reached.

% Select some (positive) pc other than zero for the simulation ...
    to run
% until differences of pc% appear between the analytical and the ...
    numerical
% solution. Example: pc = 10 means that the simulation will run ...
    until
% discrepancies of 10% show up.

tf = zeros(5,1); Rfnum = zeros(5,1); Rfanal = zeros(5,1);
% Try different values of the integration parameter beta.
for ibeta = 0:4
    beta = 0.25*ibeta;
    [tvec,Rnum,Ranal,trho,rhomat] = funPW3(N,R0,rho0,beta,dt,pc);

    figure(ibeta+1)
    plot(tvec,Rnum,'o',tvec,Ranal); xlim([0 0.7]); ylim([0, ...
        R0]); grid
    xlabel('t [s]'); ylabel('R [cm]'); ...
        legend('Numerical','Analytical')
    title(['Radius of the star (\beta = ' num2str(beta) ')'])
    figure(ibeta+6)
    semilogy(1:N,rhomat)
    grid; xlabel('i'); ylabel('\rho_{i+1/2}');
    title(['Density profile at different times (\beta = '...
        num2str(beta) ')'])
    tf(ibeta+1) = tvec(end);
    Rfnum(ibeta+1) = Rnum(end);
    Rfanal(ibeta+1) = Ranal(end);
end

% Function which performs the free fall collapse simulation.
function [tvec,Rnum,Ranal,trho,rhomat] = ...
    funPW3(N,R0,rho0,beta,dt,pc)

% Constants.
G  = 6.67259*10^-8;       % Gravitational constant in [cm^3 g^-1 ...
    s^-2].
M = 4*pi/3*R0^3*rho0;     % Total mass in [g].
dm = M/N;                 % Mass difference between consecutive ...
    shells.
mvec = dm*(0:N);          % LENGTH N+1. Thus, m_1 = 0 and ...
    m_{N+1} = M.

% Initial conditions.
rhovec = rho0*ones(1,N);
```

```
uvec = zeros(1,N+1);
rvec = zeros(1,N+1);
for i=1:N
    rvec(i+1) = (3*dm/(4*pi*rhovec(i))+rvec(i)^3)^(1/3);
end

tvec(1) = 0;
Rnum(1) = R0;
Ranal(1) = R0;

% Time loop.
time = 0; it = 0; saverho = 0; Δ = zeros(3,N);
while dt>1e-6
    time = time+dt;
    it = it+1;

    uvecn = uvec;  % So as to update uvec without modifying the ...
        previous time step velocities.
    rvecn = rvec;  % So as to update rvec without modifying the ...
        previous time step radii.
    rhovecn = rhovec;

    % While loop (epsilon) and update (x+dx).
    epsilonrho = 1; epsilonu = 1; epsilonr = 1;
    while epsilonrho > 1e-6 || epsilonu > 1e-6 || epsilonr > 1e-6

        % Internal shell
        C1 = 1/rhovec(1)-4/3*pi*rvec(2)^3/dm;
        C2 = (uvec(2)-uvecn(2))/dt+(1-beta)*G*mvec(2)/rvecn(2)^2+...
            beta*G*mvec(2)/rvec(2)^2;
        C3 = (rvec(2)-rvecn(2))/dt-(1-beta)*uvecn(2)-beta*uvec(2);
        A = [-1/rhovec(1)^2   0      -4*pi*rvec(2)^2/dm;...
             0                1/dt   -2*beta*G*mvec(2)/rvec(2)^3;...
             0                -beta  1/dt ...
                                          ];
        b = [-C1; -C2; -C3];
        dxvec = A\b;
        Δ(1,1)=dxvec(1); Δ(2,1)=dxvec(2); Δ(3,1)=dxvec(3);

        % Intermediate shells and external shell
        for j=2:N
            F1 = 1/rhovec(j)-4/3*pi*(rvec(j+1)^3-rvec(j)^3)/dm;
            F2 = (uvec(j+1)-uvecn(j+1))/dt+(1-beta)*G*...
                mvec(j+1)/rvecn(j+1)^2+beta*G*mvec(j+1)/rvec(j+1)^2;
            F3 = (rvec(j+1)-rvecn(j+1))/dt-(1-beta)*uvecn(j+1)-...
                beta*uvec(j+1);
            A = [-1/rhovec(j)^2 0     -4*pi*rvec(j+1)^2/dm;...
                 0              1/dt  ...
                     -2*beta*G*mvec(j+1)/rvec(j+1)^3;...
                 0              -beta 1/dt ...
                                                ];
            b = [-F1-4*pi/dm*rvec(j)^2*Δ(3,j-1); -F2; -F3];
            dxvec = A\b;
        Δ(1,j)=dxvec(1); Δ(2,j)=dxvec(2); Δ(3,j)=dxvec(3);
        end

        % UPDATE VARIABLES
        rhovec = rhovec+Δ(1,:);
        uvec = [0 uvec(1,2:end)+Δ(2,:)];
        rvec = [0 rvec(1,2:end)+Δ(3,:)];

        % Check error
```

```
103             epsilonrho = max(abs(Δ(1,:)./rhovec));
104             epsilonu = max(abs(Δ(2,:)./uvec(1,2:end)));
105             epsilonr = max(abs(Δ(3,:)./rvec(1,2:end)));
106         end
107
108         % Calculate analytical solution.
109         x0 = Ranal(end);
110         Rana = newton(x0,time,1e-10,100);
111         Rana = real(Rana);
112
113         % Conditions for getting as close as possible to 1% of the ...
                initial
114         % radius.
115         if pc == 0 && rvec(end)<0.01*R0
116             it = it-1;
117             time = time-dt;
118             dt = dt/2;
119             rhovec = rhovecn;
120             uvec = uvecn;
121             rvec = rvecn;
122         elseif pc == 0 && rvec(end)>0.01*R0
123             % Save time.
124             tvec(it+1) = time;
125
126             % Save numerical solution.
127             Rnum(it+1) = rvec(end);
128
129             % Save analytical solution.
130             Ranal(it+1) = Rana;
131
132             % Save density profile once every ten time steps.
133             if mod(it,10) == 0
134                 saverho = saverho+1;
135                 rhomat(saverho,:) = rhovec;
136                 trho(saverho) = time;
137             end
138         end
139
140         % Conditions for getting as close as possible to a certain ...
                error pc.
141         if pc ≠ 0 && abs(rvec(end)-Rana)/Rana>pc/100
142             it = it-1;
143             time = time-dt;
144             dt = dt/2;
145             rhovec = rhovecn;
146             uvec = uvecn;
147             rvec = rvecn;
148         elseif pc ≠ 0 && abs(rvec(end)-Rana)/Rana<pc/100
149             % Save time.
150             tvec(it+1) = time;
151
152             % Save numerical solution.
153             Rnum(it+1) = rvec(end);
154
155             % Save analytical solution.
156             Ranal(it+1) = Rana;
157
158             % Save density profile once every ten time steps.
159             if mod(it,10) == 0
160                 saverho = saverho+1;
161                 rhomat(saverho,:) = rhovec;
162                 trho(saverho) = time;
```

```
        end
    end
end
end

function R = newton(x0,t,tol,kmax)
k=0; tolk=1; X=x0;
while tolk>tol && k<kmax
    m=length(x0);
    I=eye(m);h=sqrt(eps);
    for j=1:m
        f1=feval(@analytic,x0-I(:,j)*h,t);
        f2=feval(@analytic,x0+I(:,j)*h,t);
        DF(:,j)=(f2-f1)/(2*h);
    end
    Fk=feval(@analytic,X(:,k+1),t);
    Δxk=DF\(-Fk);
    xk=X(:,k+1)+Δxk;
    X=[X xk];
    tolk=max(abs(X(:,k+1)-X(:,k+2)));
    k=k+1;
end
R = X(:,end);
end

function F = analytic(Ranalyticalpc,t)
R0 = 1e4; rho0 = 1e7; G  = 6.67259*10^-8;
F = (sqrt(1-Ranalyticalpc/R0).*sqrt(Ranalyticalpc/R0)+...
    asin(sqrt(1-Ranalyticalpc/R0)))/sqrt(8*pi*G*rho0/3)-t;
end
```

References

[1] Analytical solution. `http://www.astro.uu.se/~hoefner/astro/teach/apd_files/apd_collapse.pdf`.

[2] Jordi José. *Stellar Explosions: Hydrodynamics and Nucleosynthesis*. 2015.