A numerical approach to spherically-symmetric collapse

Andrew Zeng

Contents

1 Introduction

The goal of this project is to simulate the gravitational collapse of a spherically-symmetric dust cloud using techniques from general relativity. Specifically, we will use numerical methods to simulate the collapse of a ball of dust; This is viewed as a rough approximation of the real-life collapse of stars into black holes.
Project link

2 Summary of General Relativity

Einstein's theory of General Relativity has been hugely successful in predicting real-life astrophysical phenomena. We give a brief overview of the theory below.

2.1 The geometry of spacetime

At its core, general relativity posits that spacetime is a four-dimensional manifold, with a special pseudo-Riemannian metric. By a manifold, we mean that locally, each point in spacetime resembles Euclidean space. There are four dimensions due to the three spatial and one temporal dimension. This manifold is equipped with a smoothly-varying dot product at each tangent space, called the **metric**. To be more precise, the metric is a tensor field, and its value at each point is the dot product for the tangent space at that point.
As we will see, the metric essentially contains all the information we need to make calculations about this manifold. In particular, the metric allows us to derive a notion of curvature on this manifold. This is encapsulated in the Riemann curvature tensor $R^{\mu}_{\nu\sigma\rho}$, which can be calculated from just the metric and its inverse.
Einstein discovered a connection between the matter and energy present in spacetime, and its curvature, which is encapsulated in **Einstein's equation**.

2.2 Einstein's equation

Einstein's equation relates the geometry of spacetime to the matter and energy present in it. It is

$$G_{\mu\nu} = 8\pi G T_{\mu\nu} \tag{1}$$

On the left, we have the Einstein tensor, a tensor field which is built up from only the metric and its partial derivatives. It contains information about the geometry of the space we are working in. On the right, we have the constant $8\pi G$ (where G is Newton's gravitational constant) and the stress-energy tensor $T_{\mu\nu}$, which contains information about the matter/energy present at a point and its momentum. Although it looks simple, Einstein's equation is actually a complicated, highly nonlinear set of 10 coupled partial differential equations in the components of the metric. In common parlance, "solving Einstein's equations" either means starting with a matter distribution (the $T_{\mu\nu}$) and solving for a metric $g_{\mu\nu}$ that produces the correct $G_{\mu\nu}$ that equates the two sides. Due to the highly non-linear regime that we are working in, this is a difficult task. In fact, Einstein himself thought that the equations would never be solved. However, it is possible to obtain solutions if certain symmetry assumptions are made. For example, Einstein was proven wrong just a month after he published his theory of general relativity by Schwarzschild, who published his solution to Einstein's equations. His solution describes the space surrounding a static, spherically symmetric object:

$$g_{\mu\nu} = \begin{bmatrix} 1-\frac{R_s}{r} & 0 & 0 & 0 \\ 0 & \frac{1}{1-\frac{R_s}{r}} & 0 & 0 \\ 0 & 0 & r^2 & 0 \\ 0 & 0 & 0 & r^2\sin^2\theta \end{bmatrix} \tag{2}$$

Where we are working in units where $c = 1$, and R_s is the Schwarzschild radius of the spherical object (the radius needed to compress the object to in order to transform it into a black hole).

2.3 Relation to this project

As we may find metrics that describe static scenarios, we may also find metrics that describe a situation that is changing through time. For example, the Kerr metric describes a rotating black hole, and the four-dimensional de-Sitter space solution describes a universe that is expanding in time (in fact, this metric is consistent with the observed acceleration of the expansion rate of the universe, so it is used in many cosmological models).
The situation we have on hand in our project is that of gravitational collapse: objects will tend to collapse under their own weight due to the force of gravity unless there is a force counteracting this. Stars are no exception: they are only prevented from collapsing from the pressure created by the nuclear reactions occurring in their interior. However, the fuel necessary to complete these reactions is of finite quantity, and when it runs out, the star will begin to collapse under its own weight.

In this project, we aim to numerically recover the metric $g_{\mu\nu}$ that describes a symmetric sphere of matter undergoing gravitational collapse. Note that this is a recursive process: starting with the matter field described by $T_{\mu\nu}$, we numerically solve for the metric $g_{\mu\nu}$, which then can be used to update the state of the matter (its location and velocity), which then produces a new $T_{\mu\nu}$, which means we have to re-solve for the metric, and so on. This topic has been broached by many physicists before, and we now give a brief overview of their contributions.

3 Previous work

3.1 Oppenheimer-Snyder

The first attempt at tackling the problem of finding a metric that described gravitational collapse was from Oppenheimer and Snyder, who in 1939 published a metric that described the geometry both inside and outside the collapsing star. Their solution made many reasonable simplifying assumptions, such as spherical symmetry and uniform conditions inside the star. Their approach was to start with a general metric that obeyed spherical symmetry,

$$ds^2 = e^{\nu}dt^2 - e^{\lambda}dr^2 - r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{3}$$

where ν and λ are functions of both t and r. Then, they calculated the Einstein tensor of this metric and used the appropriate $T_{\mu\nu}$ that described a spherically symmetric mass to solve for the functions ν and λ. [2] They obtained a metric tensor that depended on whether the point is inside or outside the collapsing star. Outside the star, they found that the space could be described by the Schwarzschild metric. This is unsurprising, given their initial assumption that the collapsing star is spherically symmetric. Inside the star however, they found that space is described by the Friedmann-Robertson-Walker (FRW) metric. Curiously, this metric can also be used to describe an expanding, isotopic universe. Our task at hand is to replicate Oppenheimer and Snyder's efforts using the methods of numerical analysis.

4 Deriving the equations of motion

The geometry of spacetime is wholly determined by the *metric*, which is a symmetric positive-definite rank (0,2) tensor that takes a value at every point in spacetime. In our case, spacetime will consist of three spatial and one temporal dimension (3+1). From a mathematical point of view, the metric is a smoothly-varying way of taking inner products on tangent spaces throughout our spacetime, which can be thought of as a manifold. Different metrics model different setups and situations; For example, the metric of flat Minkowski space is

$$ds^2 = -dt^2 + dx^2 + dt^2 + dz^2 \tag{4}$$

The metric we concern ourselves with will be a general spherically-symmetric metric, which can be written as

$$ds^2 = -e^{a(r,t)}dt^2 + e^{b(r,t)}dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{5}$$

Using the SymPy Python package, the Einstein tensor for this spacetime was calculated to be

$$G_{\mu\nu} = \begin{bmatrix} \frac{e^{\alpha-\beta}}{r^2}(r\beta' + e^\beta - 1) & \frac{1}{r}\dot{\beta} & 0 & 0 \\ \frac{1}{r}\dot{\beta} & \frac{1}{r^2}(r\alpha' - e^\beta + 1) & 0 & 0 \\ 0 & 0 & \Delta & 0 \\ 0 & 0 & 0 & \Theta \end{bmatrix} \tag{6}$$

Where

$$\Delta = \frac{r}{2}(\frac{r}{2}e^\alpha(\alpha')^2 - \frac{r}{2}e^\alpha\alpha'\beta' + re^\alpha\alpha'' + \frac{r}{2}e^\beta\dot{\alpha}\dot{\beta} - \frac{r}{2}e^\beta\dot{\beta}^2 - re^\beta\ddot{\beta} + e^\alpha\alpha' - e^\alpha\beta')e^{-\alpha-\beta} \tag{7}$$

$$\Theta = \frac{r}{2}(\frac{r}{2}e^\alpha(\alpha')^2 - \frac{r}{2}e^\alpha\alpha'\beta' + re^\alpha\alpha'' + \frac{r}{2}e^\beta\dot{\alpha}\dot{\beta} - \frac{r}{2}e^\beta\dot{\beta}^2 - re^\beta\ddot{\beta} + e^\alpha\alpha' - e^\alpha\beta')e^{-\alpha-\beta}\sin^2\theta \tag{8}$$

Where $'$ denotes differentiation with respect to r, and $\dot{}$ denotes differentiation with respect to t. We now have to solve Einstein's field equations

$$G_{\mu\nu} = 8\pi G T_{\mu\nu} \tag{9}$$

in order to recover the functions α and β. Because we are working with a perfect fluid, we have

$$T_{\mu\nu} = (p + \rho)u_\mu u_\nu + pg_{\mu\nu} \tag{10}$$

Since we are working with dust, a particular type of perfect fluid, we have $p = 0$, so the above simplifies to

$$T_{\mu\nu} = \rho u_\mu u_\nu \tag{11}$$

Where ρ is the density and u_μ is the four-velocity.

4.1 Overview of the procedure for obtaining the equations of motion

Given an initial matter distribution (a sphere in our case), we will have the initial value for u_μ at all points in our initial space. In our code, we set $u_t = 1$ and $u_r = u_\theta = u_\phi = 0$. Hence initially, the "star" is at rest spatially and it is only moving forward in time.
This will give us the stress-energy tensor. Given $T_{\mu\nu}$, we can use the Einstein equation $G_{\mu\nu} = 8\pi G T_{\mu\nu}$ to numerically approximate the values of $a(r, t_0)$ and $b(r, t_0)$ at each point; Given $a(r, t_0)$ and $b(r, t_0)$ at each point, we obtain the metric $g_{\mu\nu}$ at each point. Now that we have the metric, we may use it to find the Christoffel symbols, which gives us the geodesic equation

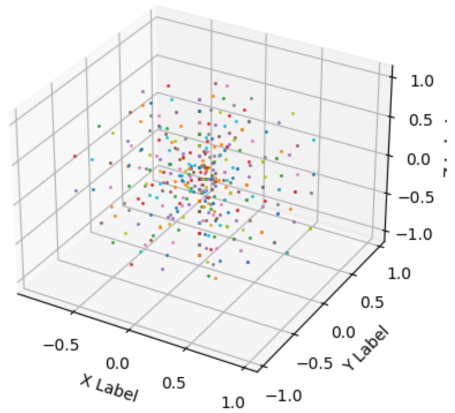
$$\frac{d^2 x^\mu}{d\lambda^2} + \Gamma^\mu_{\nu\sigma} \frac{dx^\nu}{d\lambda} \frac{dx^\sigma}{d\lambda} = 0 \tag{12}$$

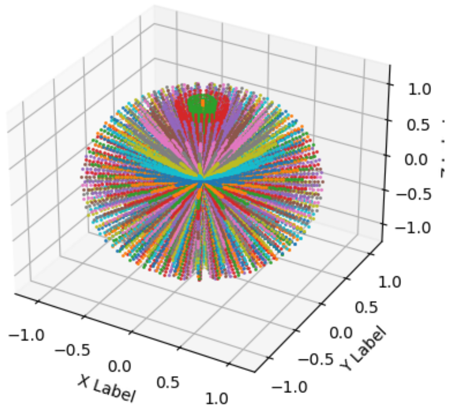
Here, $x(\lambda)$ is a path through space-time parameterized by λ. We solve the geodesic equation numerically, and this tells us how matter (dust in our case) evolves in space. This allows us to time-step the simulation forward.

4.2 Numerical procedure

4.2.1 Spherical mesh

An unconventional approach we took for tackling this project is to use a spherical mesh whose grid points served as pseudo-dust points. We will explain what this means momentarily. Our spherical mesh is composed of concentric spheres centered around the origin, each of which has rings of grid points on it. We display two meshes below. The left mesh is a reduced version of the full mesh, shown on the right:





Now, we explain what it means for these grid points to serve as pseudo-dust points. This is a crucial part of our simulation. At every step in our simulation, we want to approximate the values of $a(r, t)$ and $b(r, t)$, among other quantities, at each of our dust points. To do this numerically however, is only tractable if the dust points in our simulation lie on regular intervals and straight lines. This way we can derive numerical schemes to obtain the desired quantities. Another possibility is to use interpolation methods, but we did not develop that idea further.

In simulations however, it is obvious that we cannot expect the dust particles to lie on such nice intervals, since they are constantly in motion. However, the mesh points do lie on regular intervals.

Hence we may treat each mesh point as a "dust point" by giving each mesh point the average of the attributes of the dust points that are currently contained in a small radius around the mesh point.

In this schematic, the dotted blue point is a grid point in the mesh, shown as black lines. In order to treat this grid point as a bona fide dust point, we assign it the average u_μ, $a(r,t)$, $b(r,t)$, etc. of the dust points in a small radius around it, shown as the red circle. Hence, the green dust points will contribute to this average, while the purple ones will not. Note that the image is not to scale. In the simulation, the grid lines are denser compared to the dust particles.

Hence, this allows us to approximate the quantities u_μ, $a(r,t)$, and $b(r,t)$ on the regularly-spaced grid points.

4.2.2 Boundary conditions

On the boundary of the collapsing dust ball, we want the metric given by $a(r,t)$ and $b(r,t)$ to match the metric of ambient space, which will be the Schwarzschild metric in our case. That metric is

$$ds^2 = -(1 - \frac{r_s}{r})dt^2 + \frac{1}{(1 - \frac{r_s}{r})}dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{13}$$

Where $r_s = 2GM$ is the Schwarzschild radius of the mass. To simplify computations, we will take $r_s = 1$.

Hence on the boundary of the collapsing dust ball, we want the following two metrics to be the same:

$$ds^2 = -e^{a(r,t)}dt^2 + e^{b(r,t)}dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{14}$$

$$ds^2 = -(1 - \frac{r_s}{r})dt^2 + \frac{1}{(1 - \frac{r_s}{r})}dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{15}$$

We solve for $a(r,t)$ and $b(r,t)$ to obtain:

$$a(r_b,t) = \log\left(1 - \frac{1}{r_b}\right) \quad b(r_b,t) = -\log\left(1 - \frac{1}{r_b}\right) \tag{16}$$

for all $t >= 0$, where r_b is the current radius of the dust ball.

4.2.3 Solving for the functions a and b

The first step towards finding the equations of motion for the dust is to solve for the functions $a(r,t)$ and $b(r,t)$ on the *current time slice* t_n.

We already know the boundary conditions

$$a(r_b,t) = \log\left(1 - \frac{1}{r_b}\right) \quad b(r_b,t) = -\log\left(1 - \frac{1}{r_b}\right) \tag{17}$$

The fact that we know them is symbolized by green checkmarks in the image above. To determine the rest of the values, we will use the (t,t) and (r,r) components of the Einstein field equations:

$$G_{tt} = 8\pi T_{tt} = 8\pi\rho u_t^2 \tag{18}$$

$$\frac{e^{a-b}}{r^2}(rb' + e^b - 1) = 8\pi\rho u_t^2 \tag{19}$$

$$b' = \frac{8\pi\rho r u_t^2}{e^{a-b}} + \frac{1}{r} - \frac{e^b}{r} \tag{20}$$

$$\frac{b(r_m) - b(r_{m-1})}{\Delta r} = \frac{8\pi\rho r_m u_{t_n}^2}{e^{a(r_m)-b(r_m)}} + \frac{1}{r_m} - \frac{e^{b(r_m)}}{r_m} \tag{21}$$

$$b(r_{m-1}) = b(r_m) - \Delta r(\frac{8\pi\rho r_m u_{t_n}^2}{e^{a(r_m)-b(r_m)}} + \frac{1}{r_m} - \frac{e^{b(r_m)}}{r_m}) \tag{22}$$

This allows us to find the discretized values of $b(r,t)$ at our current time slice $t = t_n$. Note that in the last two lines above, we omitted writing t_n in the arguments for conciseness. Since we know what $a(r_b,t)$ and $b(r_b,t)$ are (on the rightmost grid point), we can use this to recover the rest of the values by "marching left".
Similarly, we may recover the values of $a(r,t_n)$ by looking at the (r,r) component of the Einstein equation:

$$G_{rr} = 8\pi T_{rr} = 8\pi\rho u_r^2 \tag{23}$$

$$\frac{1}{r^2}(ra' - e^b + 1) = 8\pi\rho u_r^2 \tag{24}$$

$$a' = 8\pi\rho u_r^2 r - \frac{1}{r} + \frac{e^b}{r} \tag{25}$$

$$\frac{a(r_m) - a(r_{m-1})}{\Delta r} = 8\pi\rho u_{r_m}^2 r_m - \frac{1}{r_m} + \frac{e^{b(r_m)}}{r_m} \tag{26}$$

$$a(r_{m-1}) = a(r_m) - \Delta r(8\pi\rho u_{r_m}^2 r_m - \frac{1}{r_m} + \frac{e^{b(r_m)}}{r_m}) \tag{27}$$

After this procedure, we have found all the values of $a(r,t_n)$ and $b(r,t_n)$ on the grid points on the fixed time slice $t = t_n$. This means we have all values of the metric

$$ds^2 = -e^{a(r,t)}dt^2 + e^{b(r,t)}dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2) \tag{28}$$

on this time slice as well, since a and b were the only unknowns. Our next step is to find the Christoffel symbols

$$\Gamma^\lambda_{\mu\nu} = \frac{1}{2}g^{\lambda\sigma}(\partial_\mu g_{\nu\sigma} + \partial_\nu g_{\sigma\mu} - \partial_\sigma g_{\mu\nu}) \tag{29}$$

We have calculated them using SymPy's symbolic differentiation package, and they involve the partial derivatives of a and b with respect to both t and r. The nonzero Christoffel symbols are

$$\Gamma^t_{tt} = \frac{1}{2}\partial_t a \quad \Gamma^t_{tr} = \frac{1}{2}\partial_r a \quad \Gamma^t_{rr} = \frac{1}{2}e^{-a+b}\partial_t b \tag{30}$$

$$\Gamma^r_{tt} = \frac{1}{2}e^{a-b}\partial_r a \quad \Gamma^r_{tr} = \frac{1}{2}\partial_t b \quad \Gamma^r_{rr} = \frac{1}{2}\partial_r b \quad \Gamma^r_{\theta\theta} = -re^{-b} \quad \Gamma^r_{\phi\phi} = -re^{-b}\sin^2\theta \tag{31}$$

$$\Gamma^\theta_{r\theta} = \frac{1}{r} \quad \Gamma^\theta_{\phi\phi} = -\frac{1}{2}\sin 2\theta \tag{32}$$

$$\Gamma^\phi_{r\phi} = \frac{1}{r} \quad \Gamma^\phi_{\theta\phi} = \frac{1}{\tan\theta} \tag{33}$$

Note that the Christoffel symbols are symmetric in their lower indices:

$$\Gamma^\lambda_{\mu\nu} = \Gamma^\lambda_{\nu\mu} \tag{34}$$

It is relatively easy to find the time derivative of b; We simply use the (t, r) component of the Einstein field equations:

$$\frac{1}{r}\dot{b} = \rho u_t u_r \tag{35}$$

To find the time derivative of a, we will use a finite difference scheme in time. However, for our initial time slice this cannot be done, unless we make a crucial assumption: the universe has been in the same state before we started our simulation. This means that for $t \leq 0$, all our variables had been the same. This means if we use the scheme

$$\dot{a}(r_m, 0) = \frac{a(r_m, 0) - a(r_m, -\Delta t)}{\Delta t} \tag{36}$$

, this assumption translates to

$$a(r_m, 0) = a(r_m, -\Delta t) \tag{37}$$

and hence

$$\dot{a}(r_m, 0) = 0 \tag{38}$$

An important note is that this does not imply $b(r_m, t_n)$ is constant for all t_n at a single fixed grid location; This is because for the next time slice, b will be calculated using the stress-energy tensor and the Einstein field equations, as detailed in the previous section. Once this is completed, we obtain values of the Christoffel symbols at all points in our current timeslice. However, note that for practical reasons we do not need to explicitly compute the Christoffel symbols. That because their only purpose in our code is for writing down the geodesic equations, and we may directly compute those instead. So we move on to the geodesic equation

$$\frac{d^2x^\mu}{d\lambda^2} + \Gamma^\mu_{\nu\sigma}\frac{dx^\nu}{d\lambda}\frac{dx^\sigma}{d\lambda} = 0 \tag{39}$$

It is straightforward to write down the four geodesic equations (one for each of $\mu = t, r, \theta\phi$). Below, $\dot{}$

will refer to differentiation with respect to the affine parameter λ:

$$\ddot{t} + \frac{1}{2}\partial_t a\dot{t}^2 + \partial_r a\dot{t}\dot{r} + \frac{1}{2}e^{-a+b}\partial_t b\dot{r}^2 = 0 \tag{40}$$

$$\ddot{r} + \frac{1}{2}e^{a-b}\partial_r a\dot{t}^2 + \partial_t b\dot{t}\dot{r} + \frac{1}{2}\partial_r b\dot{r}^2 - re^{-b}\dot{\theta}^2 - re^{-b}\sin^2\theta\dot{\phi}^2 = 0 \tag{41}$$

$$\ddot{\theta} + \frac{2}{r}\dot{r}\dot{\theta} - \frac{1}{2}\sin 2\theta\dot{\phi}^2 = 0 \tag{42}$$

$$\ddot{\phi} + \frac{2}{r}\dot{r}\dot{\phi} + \frac{2}{\tan\theta}\dot{\theta}\dot{\phi} = 0 \tag{43}$$

For example, the first equation leads to

$$\ddot{t} + \frac{1}{2}\partial_t a\dot{t}^2 + \partial_r a\dot{t}\dot{r} + \frac{1}{2}e^{-a+b}\partial_t b\dot{r}^2 = 0 \tag{44}$$

$$\frac{\dot{t}_{new} - \dot{t}}{\Delta\lambda} + \frac{1}{2}\partial_t a\dot{t}^2 + \partial_r a\dot{t}\dot{r} + \frac{1}{2}e^{-a+b}\partial_t b\dot{r}^2 = 0 \tag{45}$$

$$\longrightarrow \dot{t}_{new} = \dot{t} - \Delta\lambda(\frac{1}{2}\partial_t a\dot{t}^2 + \partial_r a\dot{t}\dot{r} + \frac{1}{2}e^{-a+b}\partial_t b\dot{r}^2) \tag{46}$$

$$\tag{47}$$

Rewriting the other three equations yields

$$\dot{r}_{new} = \dot{r} - \Delta\lambda(\frac{1}{2}e^{a-b}\partial_r a\dot{t}^2 + \partial_t b\dot{t}\dot{r} + \frac{1}{2}\partial_r b\dot{r}^2 - re^{-b}\dot{\theta}^2 - re^{-b}\sin^2\theta\dot{\phi}^2) \tag{48}$$

$$\dot{\theta}_{new} = \dot{\theta} - \Delta\lambda(\frac{2}{r}\dot{r}\dot{\theta} - \frac{1}{2}\sin 2\theta\dot{\phi}^2) \tag{49}$$

$$\dot{\phi}_{new} = \dot{\phi} - \Delta\lambda(\frac{2}{r}\dot{r}\dot{\phi} + \frac{2}{\tan\theta}\dot{\theta}\dot{\phi}) \tag{50}$$

So our procedure from here on is as follows: If we have initial values of $t, \dot{t}, r, \dot{r}, \theta, \dot{\theta}, \phi, \dot{\phi}$, we can numerically simulate this system of coupled PDEs by running it forward in time. At each grid point, we choose initial values of

$$t = 0 \tag{51}$$

$$\dot{r} = \dot{\theta} = \dot{\phi} = 0 \tag{52}$$

$$\dot{t} = 0.1 \text{ (we may shrink this value in order to increase accuracy)} \tag{53}$$

Where the second line is due to the system being initially at rest spatially. These are the evolution equations for our dust particles. Looking at the four equations, we see that quantities such as r, $\dot{\theta}$, etc. are already known at the location of the dust particle without the need for approximations. However, the quantities a, b, $\partial_r a$, $\partial_r b$, $\partial_t a$, $\partial_t b$ are not necessarily known at the locations of the dust particles; They are known at the locations of the grid points, however. Hence we must interpolate their values at the locations of the dust particles. Due to the difficulties of interpolating in three dimensions along a spherical grid, we will interpolate the values at the location of a dust particle by taking the weighted average of the relevant quantities at nearby grid points. However, there is one exception: for $\dot{b} = \partial_t b$, we know from the Einstein equation that

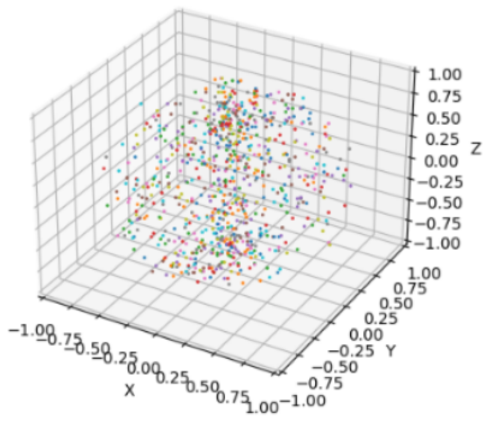
$$\frac{1}{r}\dot{b} = 8\pi\rho u_t u_r \tag{54}$$

$$\longrightarrow \dot{b} = 8\pi r\rho u_t u_r \tag{55}$$

The quantities u_t and u_r are local to the dust particle (they are its velocity in the time direction and in the radial direction respectively) hence we do not need to interpolate to obtain this value.

4.3 Implementation details

We approximate our star as a dust cloud of discrete particles. During initialization, the particles are randomly scattered in a volume defined by $r \leq 1$



This is a rough approximation; However it is the only practical choice given that the simulations are being run on a laptop.

4.3.1 Boundary detection

We would like to keep track of where the "boundary" of the star is at. This can be accomplished by marking the boundary dust particles during initialization, and keeping track of their r-coordinate as the simulation progresses.

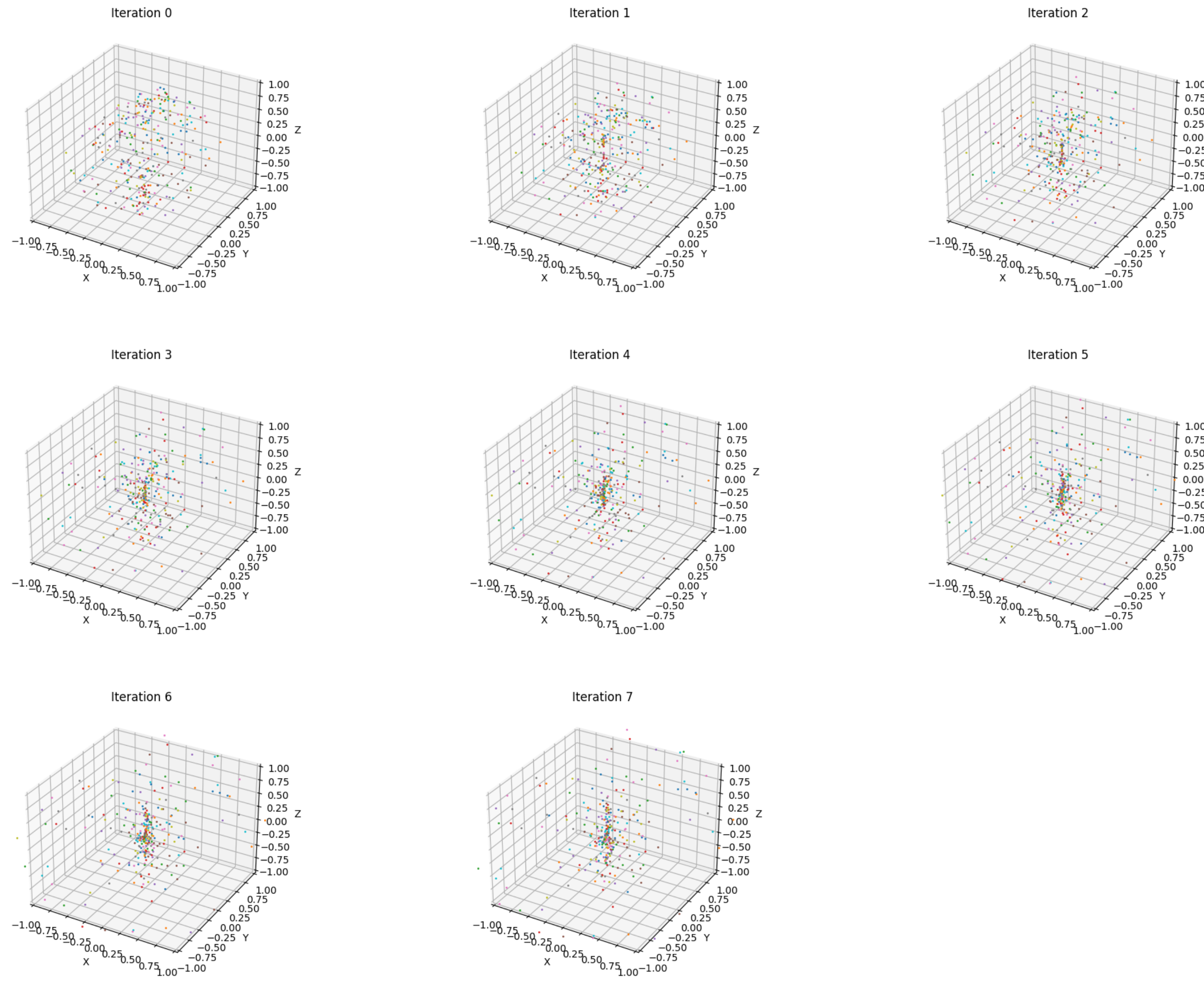
4.3.2 Calculation of the stress-energy tensor

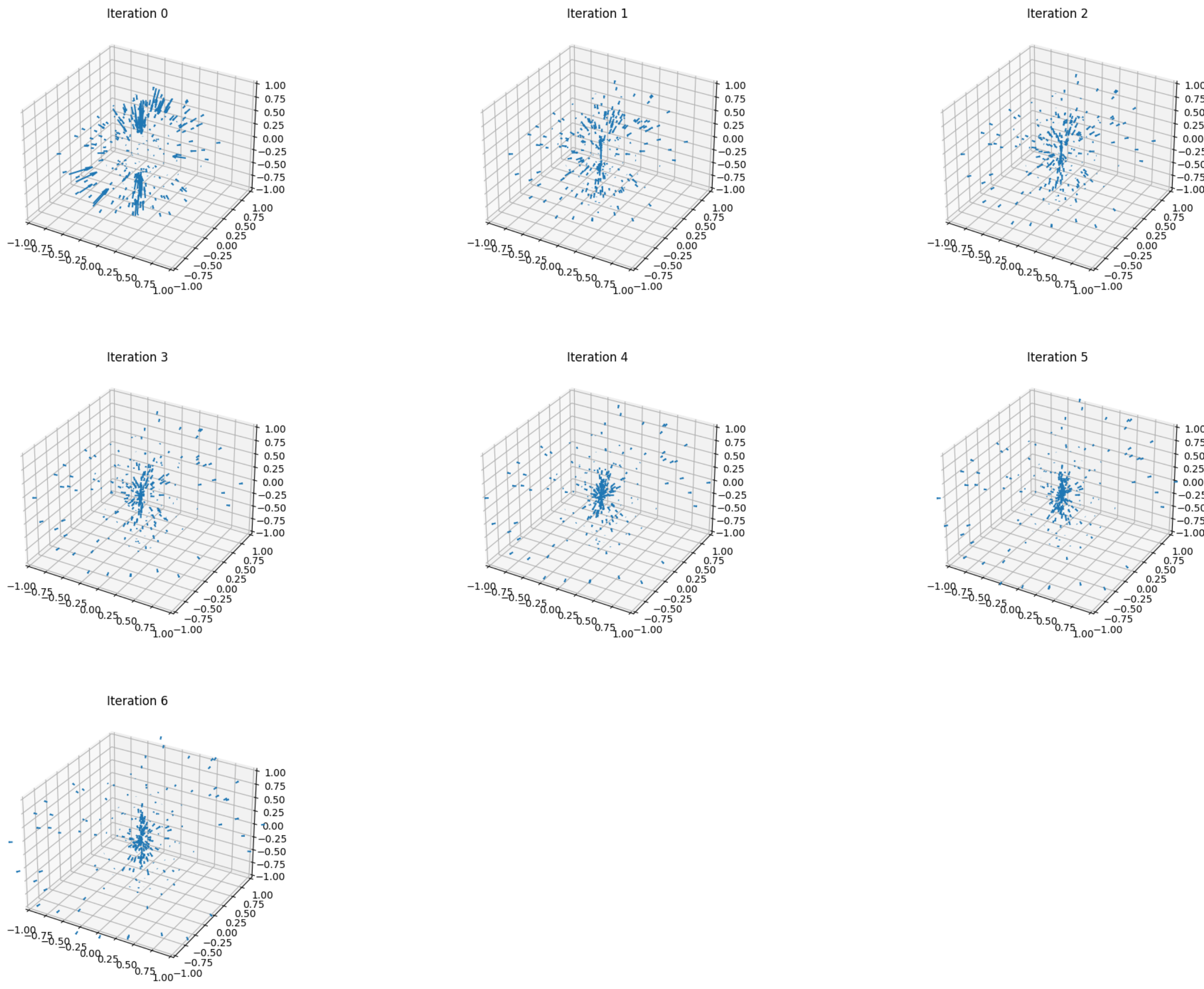
We need to approximate the value of the metric at each point occupied by a piece of dust. In order to do this, we will first calculate the metric tensor at predefined grid points, then extrapolate. However, that will require knowing the stress-energy tensor at the grid point. That can be approximated by taking a small ball around each grid point, and taking the average density ρ and velocity values u_μ in that volume. This is essentially the same step as what we did earlier in treating each grid point as a pseudo-dust point. We are now going the other way, and assigning attributes to each dust point based on the average attributes of the nearby grid points.

5 Results

We experimented with various simulation parameters, such as the number of particles, $\Delta\lambda$, and Δt. For practicality reasons, the number of particles in the dust cloud approximating the star had to be kept minimal, or else each time-step would take too long. For 500 particles, it took on average four minutes per iteration. Every iteration, we took a snapshot of the current particle locations and their velocities. The entire project is available at the following Github link:
Github link

Below we display those images taken from a sample run:





5.1 Interpretation of results

Looking at the snapshots of the dust particles' positions, we see that there is some sort of collapse occurring for the particles located closer to the origin. However, the particles near the boundary of the "star" do not follow this pattern and instead begin to move away from the origin instead. This is unexpected behaviour and there are several reasons why this may be happening:

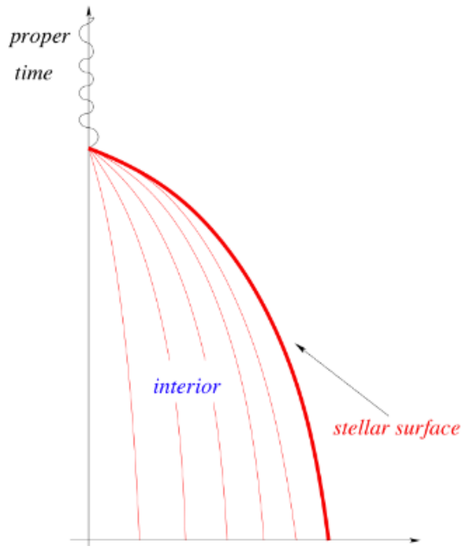
1. Numerical error: The numerical schemes we adopted in this project are mostly first-order methods. Furthermore, we used approximation methods to obtain many quantities, such as $a(r,t)$ and $b(r,t)$ for the dust particles. The efficacy of these methods can be improved by reducing the spherical mesh spacing, however this was not done due to practicality reasons.

2. Not enough particles: Our simulations relied on approximating an entire star using just a few hundred discrete dust particles. This may not emulate a star, which contains many orders of magnitude more particles (e.g. the sun has 10^{57} Hydrogen atoms), to a sufficiently high accuracy.

Another strange aspect of the simulation is that under certain simulation parameters, the dust will appear to "oscillate" between timesteps, meaning that it will contract, before expanding slightly. This was ameliorated with decreasing the Δt parameter and appeared minimally in the displayed simulation frames.

5.1.1 Comparison with the Oppenheimer-Snyder model

The Oppenheimer-Snyder model makes several experimental predictions. First, it predicts that the proper time it takes for each part of the collapsing star to reach the singularity is constant, no matter if you are close to the origin or to the boundary.



[1] We did not manage to validate this for our simulation, as we never observed the dust particles collapsing onto a single point. However, there is partial evidence of this phenomenon in our simulation in the velocity fields. In the velocity field titled "Iteration 0", it can be observed that particles closer to the boundary of the star show a greater inward velocity than the particles closer to the origin, since their velocity arrows are longer. This is consistent with Oppenheimer and Snyder's prediction that all points of the star take the same proper time to collapse, since that implies particles farther away from the origin would be moving with a greater velocity in order to reach the origin in the same time.

6 Future work

This project has demonstrated that basic simulations of spacetimes consistent with the Einstein equations may be conducted on consumer-grade machines. Possible next steps include dropping the assumption of spherical symmetry. This would involve solving for a new metric that is determined by more than two functions. Another possible step would be to decrease the spherical mesh width and increase the number of dust particles, in the hope that a more clean collapse can be observed, without the "flaking" particles near the boundary.

References

[1] Luciano Rezzolla. “An Introduction to Gravitational Collapse to Black Holes”. In: (2004).

[2] Oppenheimer; Snyder. “On Continued Gravitational Contraction”. In: *Physical Review* 56 (1939).