Black Hole Computational Modeling

Eliza Rocks, Jonah Sachs, T Thomas

October 16, 2023

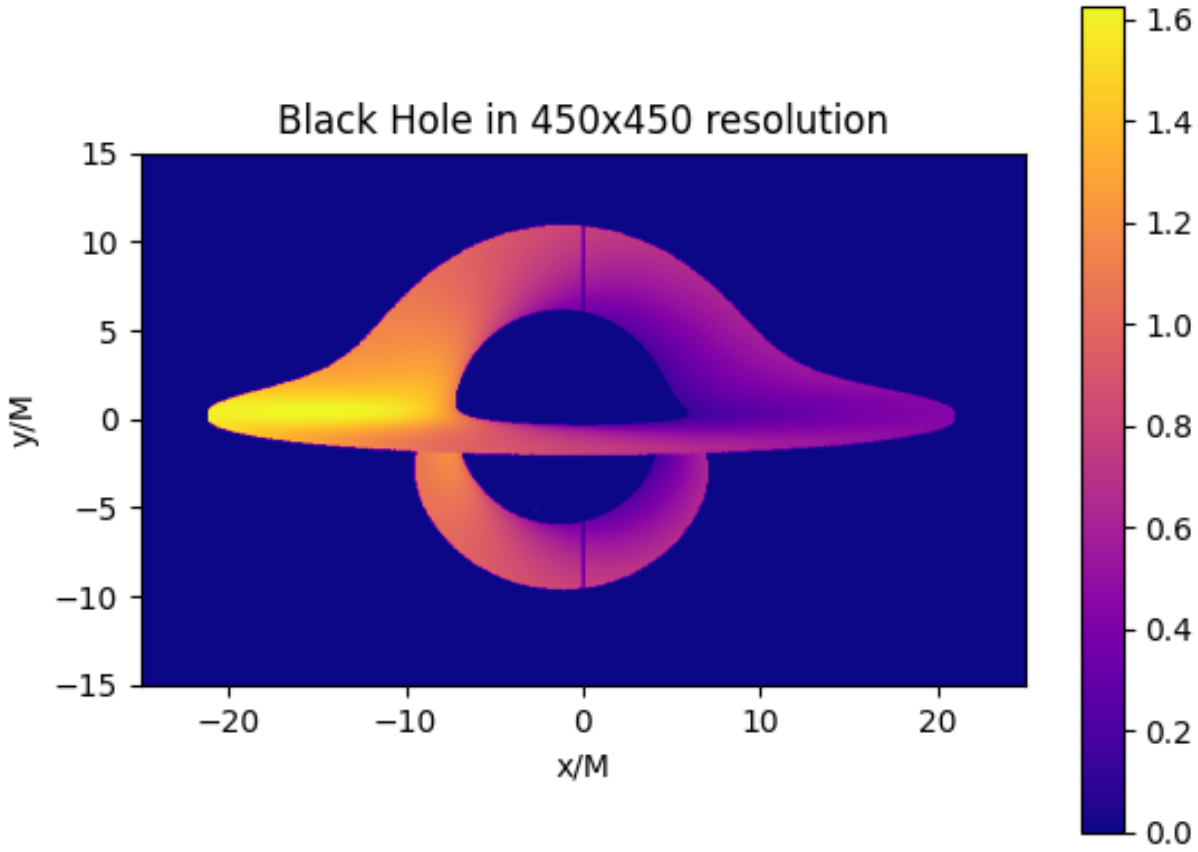


Figure 1: High Res 450 by 450 Black Hole Accretion Disk Image

1 Introduction

During this project we sought to numerically integrate for the motion of a photon in order to create a simulated image of a black hole.

The equations of motion for a photon around a black hole are:

$$\frac{dr}{dt} = \gamma^{rr}\frac{u_r}{u^t} \tag{1}$$

$$\frac{d\theta}{dt} = \gamma^{\theta\theta}\frac{u_\theta}{u^t} \tag{2}$$

$$\frac{d\phi}{dt} = \gamma^{\phi\phi}\frac{u_\phi}{u^t} - \beta^\phi \tag{3}$$

$$\frac{du_r}{dt} = -\alpha u^t \partial_r \alpha + u_\phi \partial_r \beta^\phi - \frac{1}{2u^t}(u_r^2 \partial_r \gamma^{rr} + u_\theta^2 \partial_r \gamma^{\theta\theta} + u_\phi^2 \partial_r \gamma^{\phi\phi}) \tag{4}$$

$$\frac{du_\theta}{dt} = -\alpha u^t \partial_\theta \alpha + u_\phi \partial_\theta \beta^\phi - \frac{1}{2u^t}(u_r^2 \partial_\theta \gamma^{rr} + u_\theta^2 \partial_\theta \gamma^{\theta\theta} + u_\phi^2 \partial_\theta \gamma^{\phi\phi}) \tag{5}$$

$$\frac{du_\phi}{dt} = 0 \tag{6}$$

2 Methods

We began by creating a header file titled boyer_lindquist_metric.h. In this header file we initialised all of the required variables and partial derivatives used in the equations of motion for a photon (1-6) using M, the mass and a, the spin of the black hole. Across our entire project we used M = 1.0 for simplicity; graphs are typically scaled by 1/M. The variables included $\rho, \Delta, \Sigma, \alpha, \beta^\phi, \gamma^{rr}, \gamma^{\theta\theta}, \gamma^{\phi\phi}, g_{rr}, g_{\theta\theta}, g_{\phi\phi}, g_{tt}, g_{t\phi}$. To find the partial derivatives used in equations (1-6), we first found the partial derivatives of ρ, Δ, Σ and subsequently used the derivatives to calculate the partials in the equations of motion of a photon. This way we could utilize derivative rules such as chain rule and product rule while also decreasing the computational size of calculating the necessary partial derivatives. Partial calculations were included in the header file.

The equations, along with the u^t variable were put into an equation titled dydx. The initial conditions were calculated or set and put into a vector of size 6, and a stop condition according to the test case was set. Then we used the adaptive rk4 solver with error tolerances of both 1E-12 to integrate the 6 equations using the initial conditions and stop condition. We chose adaptive rk4 for it has a nice balance of both run-time and precision. We kept our tolerances low enough to allow for a level of precision in the project, but also to make sure that we would get output files. We used a starting step size of h = 0.0001 for all cases. We followed these steps, in all of the test cases.

Finally, an output file csv was created to report the integrated r , θ , and ϕ values from a c++ vector. In the case of creating the black hole images an output file of intensity for all desired pixels was created instead.

2.1 Test Case 1: Einstein Ring in Schwarzschild Spacetime

The c++ code for finding the values for this test case can be found in photon_motion.cpp while the python code we used to graph the photon path can be found in Project_einstein.ipynb.

We began by initializing $a = 0.0$ for this case as the black hole is not spinning. We then initialized $\theta = \pi/2$, $\phi = 0$, $u_\theta = 0$ which will be true in both the case in which the photon goes around the black hole and returns to it's starting place as well as when the photon circles the black hole once before returning to it's starting place.

For the case in which the photon returns after going around the black hole r is initialized to be 10. A critical angle $\xi = 0.48857874$ is used to calculate the starting values of u_r and u_ϕ as well as the metrics, g_{rr} and $g_{\phi\phi}$, which were found using the initial values of r and theta. The stop condition is set as $\phi \geq 2\pi$.

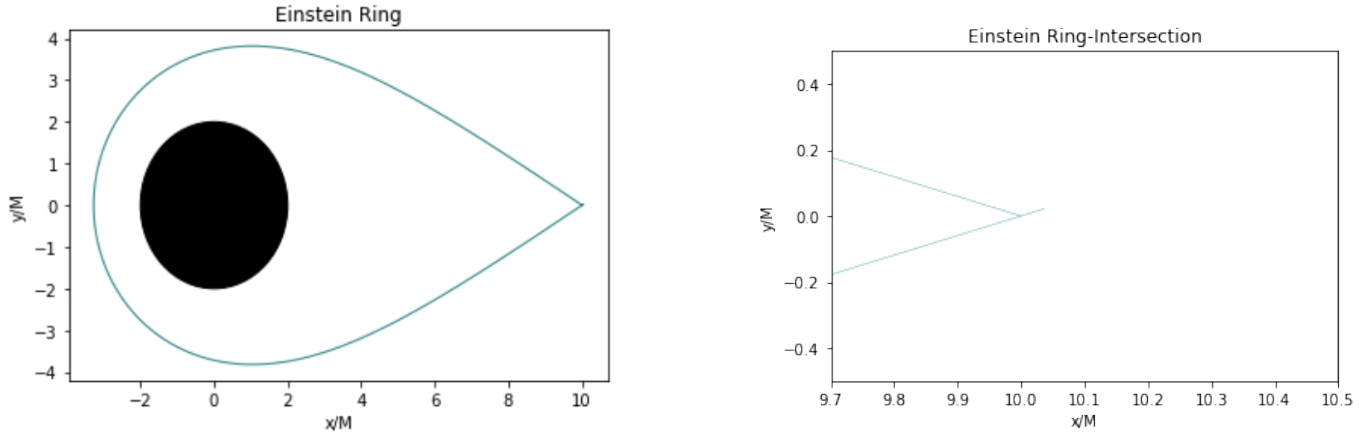


Figure 2: Graphs of photon path and error for a photon which returns after going around the black hole once.

Although close to the starting position, there is a small error in the code. Upon closer inspection we found the error distance to be 0.042629. Although small, this is still a 2.13% error compared to the location of the event horizon which in the case of the black hole not spinning is 2.

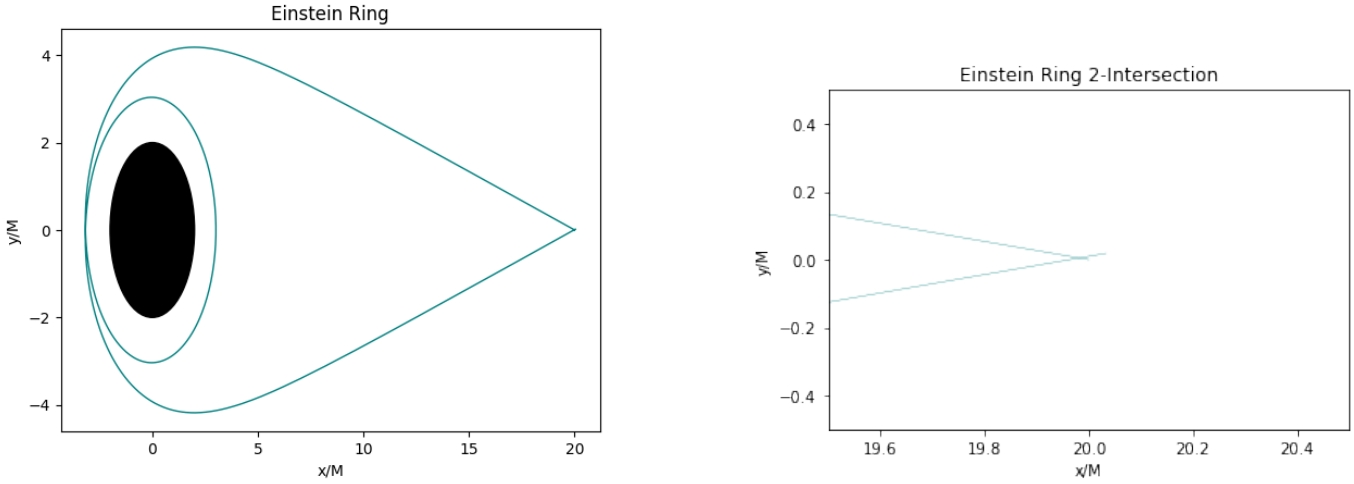


Figure 3: Graphs of photon path and error for a photon which returns after circling the black hole.

For the case in which the photon circles the black hole before returning to it's starting position r is initialized to be 20. The critical angle this time is $\xi = 0.24904964$. The stop condition in this case was set to be $\phi \geq 4\pi$. The error distance was found to be 0.037708. It is interesting that the error decreases in this case as it is 1.89% error in comparison to radius of the event horizon. However this shows that we have varying error through out calculating the photon motion throughout the project.

2.2 Test Case 2: Unstable Spherical Photon Orbits in Kerr Spacetime

The c++ code for finding the values for this test case can be found in unstable.cpp. The python code we used for plotting can be found in Error_Plotter.ipynb and Unstable_Orbit_k3d.ipynb.

We began by initializing a=1.0 for the unstable spherical orbits, which means the black hole is maximally spinning. We then initialized $\theta = \pi/2$ and $u_r = 0$. From there for each unstable spherical orbit we initialized r, u_θ, and u_{phi} as these varied between the different orbits. With these initializations we then used an RK4 adaptive solver as described above to find the photon paths of all the orbits.

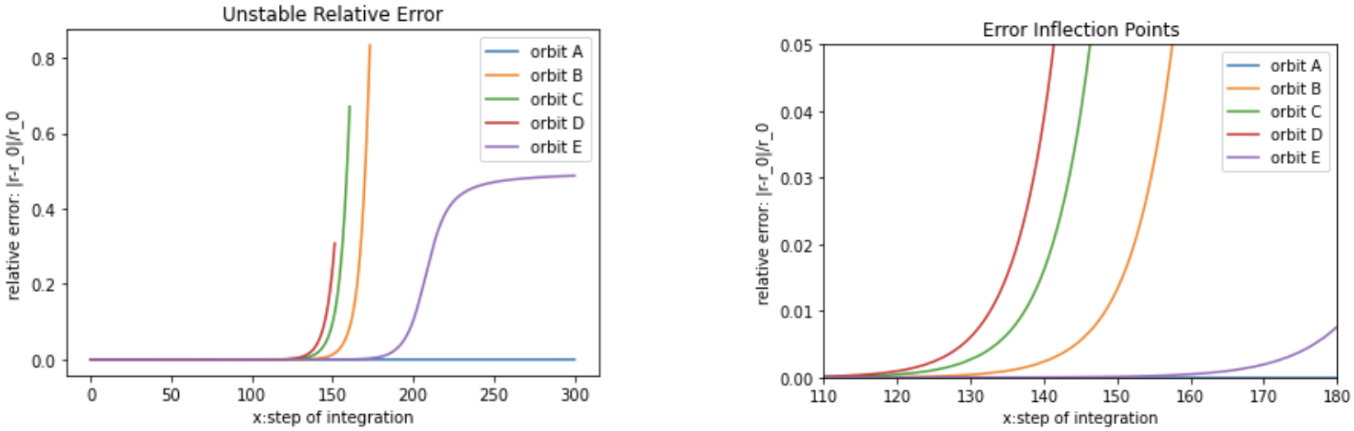
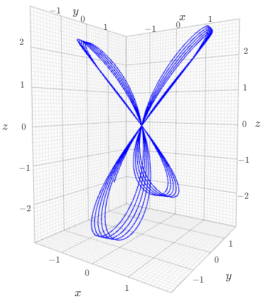


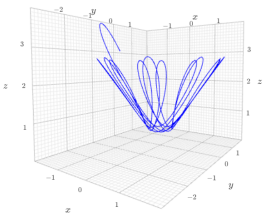
Figure 4: Graphs depicting the error for different unstable orbit conditions presented to the metric.

Analysis of the error graphs reveals that the different unstable orbit conditions remained stable for different integration periods. Orbit D degenerated the quickest, at approximately x=140. Orbits C and then B appear to have

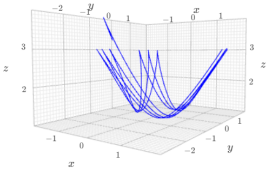
abrupt spikes in error metrics shortly after. Orbit E spikes at approximately x=210, and, given our stop condition x>300, Orbit A never degenerates and remains constant in error. The error condition provided was |r-r0 |/r0. Analysis of this equation reveals that a spike in error can only be caused by r growing towards positive or negative infinity, suggesting a photon flying away from the black hole.



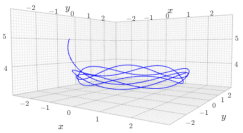
(a) k3D image of unstable orbit A



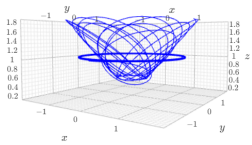
(b) k3D image of unstable orbit B



(c) k3D image of unstable orbit C



(d) k3D image of unstable orbit D



(e) k3D image of unstable orbit E

For all of the unstable orbits our stopping condition was if $x > 300$, with x being the time step that the numerical integration was on, or if $r > 5$. This allowed us to see the behavior of these orbits before they became unstable.

2.3 Producing Black Hole Images

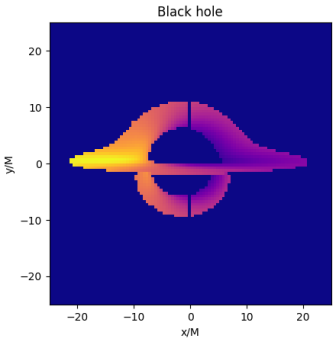
The c++ code for finding the values for this test case can be found in running_black_hole _image.cpp. The python code we used for plotting can be found in black_hole_imaging.ipynb.

We began by initializing a, rin, rout, the distance between the black hole and the camera, and resolution. We then use a double for loop to establish pixels each with a photon whose initial conditions are determined by what pixel it is in.

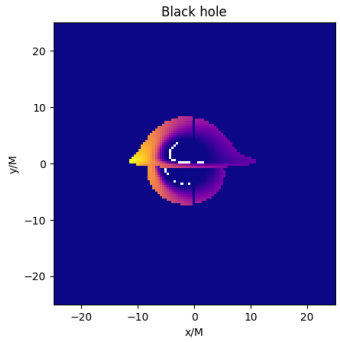
We then trace the motion of a photon backwards from the observers position. If r is greater than 1000 we assume that the photon has flown off. If r is less than the 1.01 times the event horizon then we assume that the photon has fallen into the black hole. If the photon phi component is within a tolerance of pi/2 then we assume that the photon is in the accretion disk and we calculate the intensity of the photon gravitational Redshift and Doppler boosting. We then graph the image of the black hole by setting the intensity of light to the appropriate value.

3 Results & Discussion

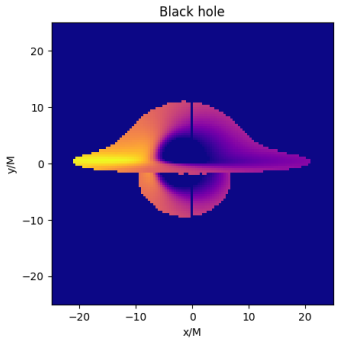
Below are images that we produced.



(a) standard accretion disk of rin = 5 rout = 20



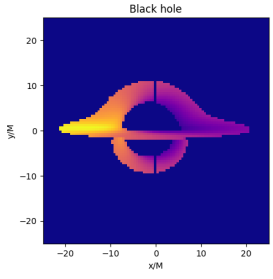
(b) Black hole image with rin=1 rout=10



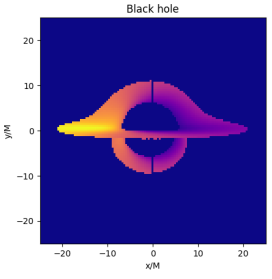
(c) Black hole image with rin=2 rout=20

Varying the rin and rout for the accretion disks changes the general shape and appearance of the disk. When we decrease the rin, you can visibly see the black hole getting smaller, however the general positions of bright and dark

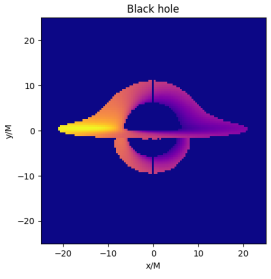
spots stays the same. Additionally, decreasing both rin and rout results in an overall smaller accretion disk, and we can see what might be the photon ring.



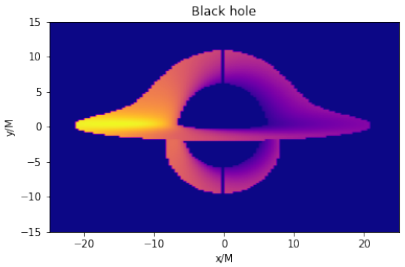
(a) standard accretion disk of a = 0.99



(b) Black hole image with a=0.5

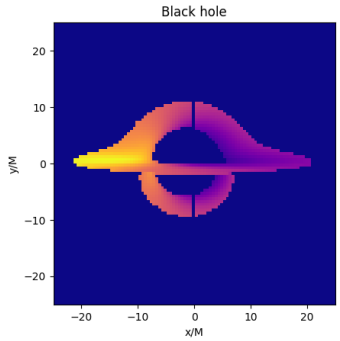


(c) Black hole image with a=0.1

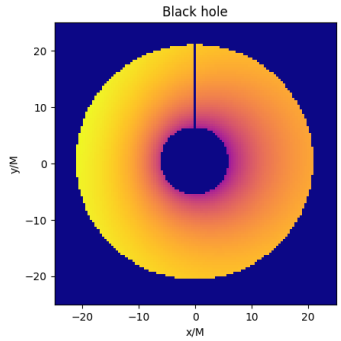


(d) Black hole image with a=0.00001

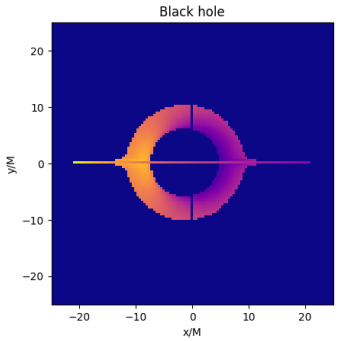
When we decrease spin it's hard to note visible differences. However if you look close enough you can see a slightly more intense difference between light and dark spots for higher spins. This makes sense because with higher spin comes more intense Doppler boost.



(a) standard accretion disk of $\theta_0 = 85$



(b) Black hole image with $\theta_0 = 30$



(c) Black hole image with $\theta_0 = 90$

When we vary the θ_0 we are able to view the black hole from different vantage points. When θ_0 is 90 degrees we get a direct side-view, meanwhile θ_0 being 30 degrees gives nearly a birds-eye view. θ_0 of 85 degrees allows us to see the different features of the accretion disk well.

3.1 Error

In our plots you can see a line on the north pole likely due to our implementation of the equations or some other form of error.

Throughout our calculations there are a few different points where error likely took place. First, we used doubles for almost all of our numbers and so there's an inherent limit to our precision, with machine precision being about 10^{-16}. Additionally, round-off error limits our accuracy. On the non-technological side, adaptive RK4 has 5th order error and our implementation of RK4 or calculations of the equations we used might not be perfect.

For all uses of adaptive RK4, both our relative and absolute tolerances were 10^{-12} and when it came to choosing that a photon hit the accretion disk our tolerance was 0.01.

The lack of a very clear photon ring in a lot of is indicative of the fact that there is large enough error to impact our results. In the future if we wanted to improve this error we could re-check our code for any typos, enforce stricter tolerances, or use a more accurate numerical integration scheme. Particularly, it might be good to use a symplectic method in order to better conserve momentum.