Accretion Disk Geodesics in Extreme Kerr Geometries

A Thesis
Presented to
The Division of Mathematics and Natural Sciences
Reed College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts

Carl L. Rodriguez

May 2010

Approved for the Division
(Physics)

_____

Joel S. Franklin

Acknowledgments

I would like to thank my parents for helping me get to and through Reed when they sure as hell didn't have to. It seems to have worked out rather well. I'd also like to thank all the people at Reed who helped me get through it here. From physics pub night to random Portland shenanigans, it's been a blast. Lastly, I want to thank the professors at Reed who shoved my interest in astrophysics and general relativity in a useful direction (namely Johnny Powell and Tom Wieting), and in particular Joel Franklin, who has endured a massive barrage of queries (both mundane and insane) all year with ridiculously good humor.

Preface

This preface is designed to give a non-technical overview of the technical aspects of this thesis. Without a background in mathematics, computer algorithms, and general relativity, some of the methodology of this thesis will seem esoteric at best. However, the conceptual ideas and results are easily accessable without the mathematics, and can be readily understood with a quick primer.

The Newtonian theory postulated that gravitation was an instantaneous effect between two objects. If the two objects had mass, then there would be a force of attraction between the two, which decreased as the distance between them increased. While accurate for most applications, discoveries at the turn of the 20^{th} proved that Newton's model was inaccurate. The response to this was Einstein's theory of general relativity: it explained several long-standing puzzles from classical physics, and replicated all the results of Newton's gravitational force. However, in general relativity, there *is* no gravitational force. Newton's spooky action at a distance was replaced by the notion of a four-dimensional universe, where space and time became a single, unified object. Furthermore, this new spacetime is no longer a rigid set of meters and clocks we draw on the world. It is a dynamic fabric; it can bend, warp, ripple, and otherwise contort itself in response to the objects residing within it. Simply put, a piece of mass will curve the spacetime around it. Normally we assume that when no forces act on a particle, it will either remain stationary, or if it was moving in a straight line, that it will continue to move in a straight line. But when the spacetime the line is moving through starts to curve, the line has no choice but to curve with it. If a planet is curving spacetime towards it, and a smaller moon passes by, then the line the moon was moving on will get curved along with the rest of the spacetime surrounding the planet, and the moon will appear attracted towards the planet.

This is precisely what this thesis aims to do: take the mathematical description of a straight line in a curved spacetime and use a computational algorithm to compute the path particles will take in the presence of the gravitational "pull" of a large, central body. Once we complete the initial program, we then aim to use our simulation to tackle a couple of problems: the first is a well known historical problem, the precession of the perihelion of Mercury. Over time, the orbit of the planet Mercury was observed to slowly change position in the sky, in a way that differed slightly from the amount predicted by Newtonian gravity. Using general relativity and the known values for the Mercury/Sun system, we will reproduce this calculation.

The second problem we wish to tackle is a poorly understood problem from modern astrophysics. In a binary star system, if the two objects come within a certain distance

of one another, the more massive of the two stars will begin to pull gas off of the smaller star and capture it. This process leads to the formation of an accretion disk, a gaseous cylinder that is slowly rotating around the larger of the two stars until it eventually falls in. This type of behavior also occurs in systems where the more massive body is actually a black hole. In general relativity, when a black hole rotates (as most are expected to), it curves the spacetime around it in such a way that anything near the hole will feel a gravitational "force" pulling it in the same direction that the black hole is rotating. If an object orbiting the black hole is traveling in the same direction as the spin, the object will speed up. If it is orbiting in the opposite direction, the object will slow down and eventually reverse direction. This is a completely new phenomenon for relativistic gravity, and the effect that it has on these accretion disks is far from clear. We want to generate a broad first pass simulation, hopefully yielding some idea of what exactly happens near the center of a black hole system.

This thesis is not about a rigorous comparison of theory to experimental fact or creating new theoretical predictions. Rather, the idea is to use a computer simulation to solve equations for gravitational motion that cannot be solved by hand. Our aim is to demonstrate known theoretical results in the context of a much "rougher than ideal" real world system. In the process, we hope to demonstrate conceptual results of black hole physics. With any luck, the thesis should be of interest to readers both technical and otherwise.

Table of Contents

List of Tables

List of Figures

Abstract

Here, we propose to develop a numerical technique for studying the motion through the curved spacetime of general relativity. Specifically, we write a computer simulation using the relativistic Lagrangian and the well known Runge-Kutta algorithms to generate geodesics for specified initial conditions. The simulation was written using the Kerr Lagrangian, which can be used to describe the gravitational effects from any spherical central body. We then applied this technique to two examples from relativistic astrophysics: the precession of the perihelion of Mercury, and the dampened transfer of mass in an accreting binary system. We were able to extract the perihelion precession to within 0.09% of the measured value. We were also able to create an accretion disk simulation, which exhibited the expected behavior in geodesic density around the central body, while respecting the ISCO for Kerr. We also discovered an unusual periodicity in the accretion disk geodesics which we were unable to explain.

Introduction

As with most papers and theses about subjects in general relativity, this introduction will focus primarily on the notations of the subject. Those readers familiar at all with general relativity can easily skip this section. Simply note that for this thesis, we will be using a $(-, +, +, +)$ signature for the metrics and four-vectors.

Vectors and One-Forms

In non-relativistic physics the vector is the starting point for all three dimensional analysis. In the case of mechanics, the vector of interest is a three dimensional position vector parameterized by time, IE

$$\vec{w}(t) = x(t)\hat{x} + y(t)\hat{y} + z(t)\hat{z} = w^a(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}$$

Where the index, a, runs from one to three, representing each of the components in turn. In relativity, we introduce the notion of a four-vector, which specifies the time coordinate along with the three spatial coordinates. We can multiply the time component by c, the speed of light, to ensure the coordinates are all in the same units (meters). Implicitly, we have set $G = c = 1$ throughout this thesis, so any time and mass values are written in terms of meters.[1] Of course, with time as part of the position vector, we need a new term to parameterize any curve in our four-dimensional spacetime. Although any parameter will do, we can use the proper time of the particle, τ, such that the four-vectors take the form

$$\mathbf{w}(\tau) = w^\mu(\tau) = \begin{pmatrix} t(\tau) \\ x(\tau) \\ y(\tau) \\ z(\tau) \end{pmatrix}$$

Letting the Greek indices run from 0 to 3 (where 0 is the time component), and the Latin indices from 1 to 3, we have in hand a vector to describe the positions of events in four-dimensional spacetime.

Once we have the terminology of a vector space, the natural next step is to define the inner product over that vector space. In three-space, the inner product is simply the sum of the products of the components, that is

[1] Or in terms of the solar mass, where 1 $M_\odot = 1476.7$ m.

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{3} a^i b^i = a^1 b^1 + a^2 b^2 + a^3 b^3$$

To extend this notion to the hyperbolic geometry of special relativity, we perform the same procedure on four-vectors, this time with a minus sign in front of the temporal component

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} = \sum_{\mu=1}^{3} a_\mu b^\mu &= a_0 b^0 + a_1 b^1 + a_2 b^2 + a_3 b^3 \\ &= -a^0 b^0 + a^1 b^1 + a^2 b^2 + a^3 b^3 \end{aligned}$$

where we have moved the minus sign in the summation to the lowered index. The vectors with a lowered index are known as **one-forms**, and in special relativity they differ from the raised index **vectors** by a minus sign in the time component. In order to switch between the two, we use the metric of the manifold. For flat, special relativistic spacetime, introduce the covariant **Minkowski metric**[2]

$$g_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

By forming the product of the two vectors with the Minkowski metric, we can rewrite the inner product as

$$\mathbf{a} \cdot \mathbf{b} = \sum_{\mu=0}^{3} \sum_{\nu=0}^{3} g_{\mu\nu} a^\mu b^\nu \equiv g_{\mu\nu} a^\mu b^\nu$$

In the interests of cleaner notation, we adopt the Einstein summation convention in the third equality, whereby any repeated raised and lowered indices are implicitly summed over.

By forming the product of the metric with a vector, we can transform between that vector and its corresponding one-form as follows:

$$a_\mu = g_{\mu\nu} a^\nu$$

Introducing the contravariant metric (defined such that $g^{\mu\alpha} g_{\mu\beta} = \delta^\alpha_\beta$) lets us raise indices using a similar procedure

$$a^\mu = g^{\mu\nu} a_\nu$$

Finally, we introduce the idea of the **line element**, the distance between two points in space. For the Minkowski metric, the line element is

[2]Although represented in matrix form here, the one-forms and metrics are *not* the vectors and matrices from linear algebra. They are objects from the more general subject of multi-linear algebra, and closely correspond to the one-forms and two-forms of differential geometry.

$$ds^2 = g_{\mu\nu}(dx^\mu)(dx^\nu) = -dt^2 + dx^2 + dy^2 + dz^2$$

where the distance, ds, is an invariant quantity. If we wanted to switch to spherical coordinates, we could achieve this by introducing the flat space spherical metric,

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

and the line element for a vector with spherical spatial coordinates would be

$$ds^2 = -dt^2 + dr^2 + r^2 d\theta^2 + r^2 \sin^2\theta \; d\varphi^2$$

where the Latin indices 1, 2, and 3 now refer to the spherical coordinates r, θ, and φ. In general relativity, we generalize the spacetime metrics from flat space to manifolds whose curvature depends on the mass, energy, and momentum of the system. We will discuss these specifically in chapter 1.

Motion in a Curved Spacetime

In general relativity, free particles in a curved spacetime will follow paths known as **geodesics**. Simply put, these paths represent the straightest possible line in a curved region. A good analogy is the example of two points on a sphere. When embedded in a higher dimmensional space,the straight Euclidean line is the shortest distance between the two points; however, when we are restricted to the surface of the sphere, the arc along the surface between the two points becomes the straightest possible line, and is a geodesic of that surface. In general relativity, a geodesic is simply the straightest line connecting two points in the four-dimensional spacetime. There are three distinct classes of relativistic geodesics:

- **Timelike Geodesics** represent the paths of massive particles in free fall. Any tangent vector along the geodesic must have norm less than zero.
- **Null Geodesics** represent the path of massless particles (light) in a curved spacetime. Any tangent vector along the geodesic must have a length equal to zero.
- **Spacelike Geodesics** have tangent vectors whose norms are greater than zero. As they represent motion faster than c, they do not describe any causal particle motion.

Since we are interested in massive particle trajectories, we will focus exclusively on timelike geodesics. Since we have parameterized our four-vectors in the proper time of the particle, we can then define the tangent vector as the proper velocity

$$\frac{dx^\mu}{d\tau} \equiv \dot{x}^\mu \tag{3}$$

We know from special relativity that the norm of the proper velocity of a particle is always -1. Therefore, we conclude that any timelike geodesics must obey

$$-1 = g_{\mu\nu}\dot{x}^\mu\dot{x}^\nu$$

This "velocity squared" term is also the basis for the **relativistic Lagrangian**. Since there is no gravitational potential in general relativity, the Lagrangian for timelike geodesics becomes

$$\mathcal{L} = \frac{1}{2}g_{\mu\nu}\dot{x}^\mu\dot{x}^\nu \tag{4}$$

with the $1/2$ inserted by convention. With equation 4 in hand, we can apply the Euler-Lagrange equations

$$\frac{d}{d\tau}\frac{\partial\mathcal{L}}{\partial\dot{x}^\mu} - \frac{\partial\mathcal{L}}{\partial x^\mu} = 0 \tag{5}$$

and are led to the four equations of motion for any spacetime, so long as we know the metric. This method is straight-forward, and is what we will employ in this thesis.

Finally, we can draw an analogy between relativistic motion and Newton's second law by introducing the geodesic equation. By applying equation 5 to the Lagrangian, we come to the equation(s)

$$\frac{d^2x^\alpha}{d\tau^2} + \Gamma^\alpha_{\mu\nu}\dot{x}^\mu\dot{x}^\nu = 0 \tag{6}$$

where the second term is the Christoffel connection of the metric, defined by

$$g_{\beta\gamma}\Gamma^\gamma_{\mu\nu} = \frac{1}{2}\left(\frac{\partial g_{\beta\mu}}{\partial x^\nu} + \frac{\partial g_{\beta\nu}}{\partial x^\mu} - \frac{\partial g_{\mu\nu}}{\partial x^\beta}\right)$$

Since the Christoffel symbol clearly goes to zero for Minkowski space, equation 6 assumes the form of Newton's second law in flat space, exactly as we would hope. Although we could use the geodesic equation for our simulation, in practice calculating the Christoffel symbols for a non-symmetric metric would require at minimum as much work as extracting the equations of motion from the Lagrangian, while yielding identical results.

Chapter 1

Black Holes

This chapter describes the basic mathematical and physical aspects of black holes. We will first cover the Schwarzschild metric, which describes the spherically symmetric spacetime outside of a central massive body (equivalent to a stationary central body problem in classical mechanics). We will then discuss the Kerr metric, describing the axial-symmetric spacetime around a rotating black hole. Our point here is not to offer rigorous derivation, but rather to provide a basic description of relevant black hole physics. In both cases we will cover the geometric features of interest and the distinctive orbital properties.

1.1 Schwarzschild Black Holes

The Schwarzschild geometry was one of the first known solutions of general relativity. Discovered by Karl Schwarzschild in 1916, the metric describes the curvature of spacetime outside of a spherically symmetric massive body. While not our main point of study, the Schwarzschild metric contains two features which will be of later interest to us: the event horizon, and the precession of orbits. It also provides a much simpler introduction to the difference between physical singularities and those that are merely coordinate artifacts.

The Schwarzschild metric in spherical coordinates[1] is:

$$g_{\mu\nu} = \begin{pmatrix} -\left(1-\frac{2M}{r}\right) & 0 & 0 & 0 \\ 0 & \left(1-\frac{2M}{r}\right)^{-1} & 0 & 0 \\ 0 & 0 & r^2 & 0 \\ 0 & 0 & 0 & r^2\sin^2\theta \end{pmatrix} \tag{1.1}$$

where M is the mass of the central body. Note that as $M \to 0$ we recover the flat space metric (equation 2). The first point to notice is the two singularities which appear in the metric at $r = 0$ and $r = 2M$. The origin represents the physical singularity, the point where spacetime contracts to a single point. While interesting, the singularity is only of immediate concern to those proper time observers unfortunate enough

[1] Strictly speaking, the coordinates used in equation 1.1 are "Schwarzschild coordinates". But for all our intents and purposes the two are interchangeable.

to encounter it. For the coordinate time observers who reside far from the source (namely, us), the second singularity is the one we are primarily concerned with. It represents the **event horizon**, a spherical surface from which no timelike or null geodesic can emerge.[2] To show this we provide a couple of arguments:

The first is a Newtonian argument. For any particle, the minimum kinetic energy required to escape to spatial infinity from a gravitational body is equal to the gravitational potential at the starting point, namely

$$G\frac{mM}{r} = \frac{1}{2}mv_{escape}^2 \tag{1.2}$$

which, when solved for radius, yields

$$r = \frac{2GM}{v_{escape}^2} \tag{1.3}$$

Now when $v_{escape} = c$, equation 1.3 represents the radius of a sphere from which light will be unable to escape from a gravitational potential. That, when converted to $c = G = 1$ units, suggests that the at $r_{horizon} = 2M$ even light will be unable to achieve the needed escape velocity. While illustrative, this argument is really nothing more than a sleight of hand. For massless particles such as the photon, equation 1.2 would have reduced to $0 = 0$ in the first line. A more rigorous argument involving null geodesics will be provided for relativistic black holes in a moment.

The event horizon also marks a final stopping point for any particle as viewed far from the source. Since $g_{00} \to 0$ and $g_{11} \to \infty$ as $r \to 2M$, the Schwarzschild line element

$$ds^2 = -\left(1 - \frac{2M}{r}\right)dt^2 + \left(1 - \frac{2M}{r}\right)^{-1}dr^2 + r^2\left(d\theta^2 + \sin^2\theta\ d\varphi^2\right) \tag{1.4}$$

becomes completely dependent on dr^2 without a dt^2 term, causing any incident particle to appear to stall at the horizon for all time.[3] Despite this, the event horizon can be shown to be a singularity in name only.

There are two types of singularities we concern ourselves with in general relativity. Physical singularities occur at a point of infinite curvature, and can be shown to exist free of coordinate representation. Other singularities are coordinate dependent, and can be shown to disappear under clever coordinate transformation. A good analogy is the $\theta = 0$ point in spherical coordinates; a singularity that can be made to disappear by transformation to Cartesian coordinates. A similar procedure works for the Schwarzschild event horizon. By replacing t with a new timelike coordinate v

$$t = v \mp \left(r + 2M\log\left|\frac{r}{2M} - 1\right|\right) \tag{1.5}$$

the line element becomes

[2]The gravitational "point of no return", for the more melodramatically inclined.

[3]This effect is only observable in coordinate time. Any proper time observer will fall through the event horizon without impedance.

$$ds^2 = -\left(1 - \frac{2M}{r}\right) dv^2 \pm 2dv\ dr + r^2(d\theta^2 + \sin^2\theta\ d\varphi^2) \tag{1.6}$$

and the singularity disappears.[4] The top sign choice corresponds to points $r \geq 2M$, while the bottom choice corresponds to $r < 2M$. This choice of coordinates, known as **Eddington-Finkelstein** coordinates[5], also provides us with a better description of the event horizon. Suppose we have a photon moving along a null geodesic at $r = 2M$. If we assume motion tangent to the surface of a sphere (for which $dr = 0$), then equation 1.6 becomes

$$ds^2 = r^2\left(d\theta^2 + \sin^2\theta\ d\varphi^2\right) = 0 \tag{1.7}$$

What this means is that any photons on the event horizon can only maintain stable orbits if they continue on that sphere forever. Any closer in, the dv^2 term reappears, albeit as a positive space-like coordinate, meaning the second term must decrease to maintain equality. That means decreasing the r^2 which reintroduces the $-2dv\ dr$ term, implying infall.

The other point of interest from Schwarzschild is the notion of **precession**. Unlike Newtonian gravitation, orbits about spherical central bodies in the Schwarzschild geometry do not close back onto themselves immediately. Instead, the point of closest approach slowly rotates around the central body, causing orbits that would close in a $1/r$ potential to appear "flower shaped". This is a result of **Bertrand's Theorem**, which permits closed orbits only for r^2 and $1/r$ potentials. For orbits in Newtonian physics, we can write down an effective potential (which includes angular momentum in the energy equation) as

$$V_{Neff}(r) = -\frac{GM}{r} + \frac{l^2}{2r^2} \tag{1.8}$$

Similarly, we can write down an effective potential for Schwarzschild, taking into account both angular momentum *and* the energy change from a curving timelike component.

$$V_{Seff}(r) = -\frac{GM}{r} + \frac{l^2}{2r^2} - \frac{Ml^2}{r^3} \tag{1.9}$$

The key difference is the second term in equation 1.9, a $1/r^3$ addition. It is this additional, minor potential that is responsible for orbital precession in Schwarzschild. One of the best known examples of this effect is the **perihelion precession** of Mercury. We will further discuss (and numerically compute) this well known effect in Chapter 2.

[4]This trick only works when you already have a different coordinate system that covers the singularity in question. You can show that a singularity exists physically in nature (and therefore independently of coordinate system) by examination of the invariant $R^{\alpha\beta\gamma\delta}R_{\alpha\beta\gamma\delta}$. We will do this in the next section for Kerr black holes.

[5]Hartle, 2003, p. 258.

1.2 Kerr Black Holes

In the previous section we treated black holes as static objects in space, studying the physical properties of a spherically symmetric spacetime; however, the vast majority of astrophysical applications (neutron stars, planets, etc.) are *not* entirely stationary. They are endowed with some angular rotation, ranging from barely noticeable (such as our sun), to so fast that a point on the surface travels at a significant fraction of the speed of light (e.g. neutron stars or black holes). In Newtonian physics this distinction was a non-issue. Since a sphere is identical when viewed under any rotation, the mass distribution, $\rho(\vec{r})$, is also unchanged under rotation, and therefore invariant with respect to spin and angular momentum. Therefore, any field equation with ρ as a source (i.e. Newtonian gravity) will also be invariant to central body spin.

In general relativity, however, the situation is not so clear cut. The relativistic source term, the stress-energy-momentum tensor (the second rank tensor which acts as the source term for the Einstein field equations), contains several components which differ radically with the addition of central body angular momentum. Spin energy will appear in the τ_{00} term, angular momentum in the τ_{01}, τ_{02}, and τ_{03} terms, and any rotationally related internal stresses in the τ_{ab} stress sub-tensor. Given such drastic changes, it becomes clear that any addition of angular momentum to a central body will substantially change the nature of the spacetime. Despite this, we are still allowed to expect a couple of things:

- The metric should revert to the Schwarzschild metric for cases with zero spin, making it a generalization of Schwarzschild.
- It should revert to the flat Minkowski space of special relativity when no mass is present.
- It should also revert to flat space at $r \to \infty$, a requirement known as asymptotic flatness.

Whereas Schwarzschild's solution was published a mere three months after the advent of general relativity, the **Kerr metric** was not discovered for nearly half a century. Written in **Boyer-Lindquist** coordinates, the metric is

$$g_{\mu\nu} = \begin{pmatrix} -\left(1 - \frac{2Mr}{\rho^2}\right) & 0 & 0 & -\frac{2aMr\sin^2\theta}{\rho^2} \\ 0 & \frac{\rho^2}{\Delta} & 0 & 0 \\ 0 & 0 & \rho^2 & 0 \\ -\frac{2aMr\sin^2\theta}{\rho^2} & 0 & 0 & \left(r^2 + a^2 + \frac{2a^2Mr\sin^2\theta}{\rho^2}\right)\sin^2\theta \end{pmatrix} \tag{1.10}$$

where

$$\Delta \equiv r^2 - 2Mr + a^2$$
$$\rho^2 \equiv r^2 + a^2\cos^2\theta$$

Equation 1.10 introduces a new constant, a, which can be interpreted as the central body spin expressed in units of length. With this interpretation, the metric matches

the three criteria we discussed above. When $a = 0$, the cross terms disappear, and the diagonal terms revert to Schwarzschild. When no mass is present, the metric reverts to flat space (albeit in oblate spherical coordinates, as we will see in a moment). Finally, at spatial infinity (as $r \to \infty$) we recover the Minkowski metric (equation 2).

The most striking feature of the Kerr metric is the off-diagonal symmetric elements in g_{03}. They represent a coupling between the t and φ components, generating cross terms between the time and azimuthal coordinates of any four-vector. Physically, this means that for any test particle near a rotating central body, the spacetime itself will cause a dragging effect, essentially a "force" in the $\hat{\varphi}$ direction.[6] The second point of interest in the Kerr geometry is the horizons (plural), but before we can move on a proper discussion of coordinates is needed.

1.2.1 Boyer-Lindquist Coordinates

When Kerr first described the metric in 1963, he used two separate coordinate systems. The first was an "advanced Eddington-Finkelstein" system, so called because as $a \to 0$, the Kerr metric reduced to the Schwarzschild metric in Eddington-Finkelstein coordinates. The second was a system strongly resembling Cartesian coordinates.[7] While both systems offer distinct advantages (the obviousness of Killing vectors, easy asymptotic reduction to Schwarzschild or Minkowski spacetimes, etc.), the large number of cross terms in the metric and the difficult physical interpretations made even the most straight-forward calculations extremely tedious.

In 1967, Boyer and Lindquist introduced the system of coordinates we use here. Apart from the advantageous analogy to spherical coordinates (especially in the weak field regime), the Kerr metric contains only a single cross term when expressed this way[8], significantly simplifying calculations involving either the field equations or the line element. The consolidation of elements also makes analysis of the horizons significantly simpler. Additionally, at spatial infinity, the coordinates resemble the perturbative metrics used to approximate rotating spacetimes before Kerr's discovery. It is from that comparison that the understanding of a as the angular spin comes.

As stated above it is the asymptotic behavior that makes this coordinate system so appealing. When $M = 0$, the coordinates reduce to oblate spheroidal coordinates in Minkowski spacetime:

$$\begin{aligned} x &= \sqrt{r^2 + a^2}\sin\theta\cos\phi \\ y &= \sqrt{r^2 + a^2}\sin\theta\sin\phi \\ z &= r\cos\theta \end{aligned} \tag{1.11}$$

[6] The "force" is fictitious, and is merely a spacetime dragging effect; however, it can be shown that the effect is similar (within a factor of 4) to the potential generated by a charged, spinning sphere in electrodynamics. See Baker, 2006.

[7] Albeit with an implicitly defined distance from the origin, $x^2 + y^2 + z^2 = r^2 + a^2(1 - (z/r)^2)$. See Visser, 2007.

[8] Although we could express the Kerr spacetime in terms of a diagonal metric, doing so sacrifices the connection to the flat space spherical coordinates we are familiar with.

For most astrophysical applications (where $r \gg a$), the above equations reduce to spherical coordinates. This is what allows us to physically interpret Boyer-Lindquist coordinates as spherical coordinates: In the flat space regime (far from the source), they *are* spherical coordinates. This interpretation is what we will use for the rest of this thesis.

1.2.2 Kerr Singularities

As before, we must make the distinction between coordinate and physical singularities. While not quite as obvious as Schwarzschild, the simplifying notation used in equation 1.10 makes the solution obvious: Kerr singularities occur where $\Delta = 0$ and $\rho^2 = 0$. We discover three singularities: two from the quadratic in Δ.

$$r_{eh} = M \pm \sqrt{M^2 - a^2} \tag{1.12}$$

and one from ρ^2 at $r = 0$ and $\theta = \pi/2$. Given that equation 1.12 becomes $r = 2M$ when $a = 0$, we can assume that the Δ singularity corresponds to the event horizon, while ρ^2 corresponds to the physical singularity. We will offer a rigorous proof of this in a moment. First we must point out the spin restriction, $a^2 \leq M^2$, that is implied by equation 1.12. Although mathematically possible, black holes in nature are not expected to have spins greater than their mass, as this would lead to the creation of causality paradoxes. These "naked singularities", while interesting, are not described properly by Boyer-Lindquist coordinates, and will be ignored for the remainder of this thesis. When $a^2 = M^2$, we call the resulting spacetime an **extreme Kerr** black hole.

Ergosphere

There is a unique feature to the Kerr spacetime that does not arise for stationary black holes. In Schwarzschild, there is definitely something peculiar about crossing the event horizon: when $r < 2M$, the signs of the g_{00} and g_{11} components in equation 1.1 both switch signs, essentially making t a spatial coordinate and r the new timelike coordinate. But in Kerr, the g_{00} component switches signs before an infalling observer hits the event horizon. Specifically, solving the quadratic for $g_{00} = 0$ gives

$$r_{es} = M \pm \sqrt{M^2 - a^2 \cos^2 \theta} \tag{1.13}$$

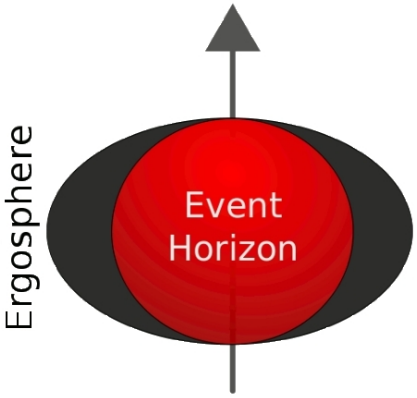

Figure 1.1: Cross section of Kerr horizons parallel to the spin axis.

which describes an ellipsoidal surface connected to the event horizon at the poles

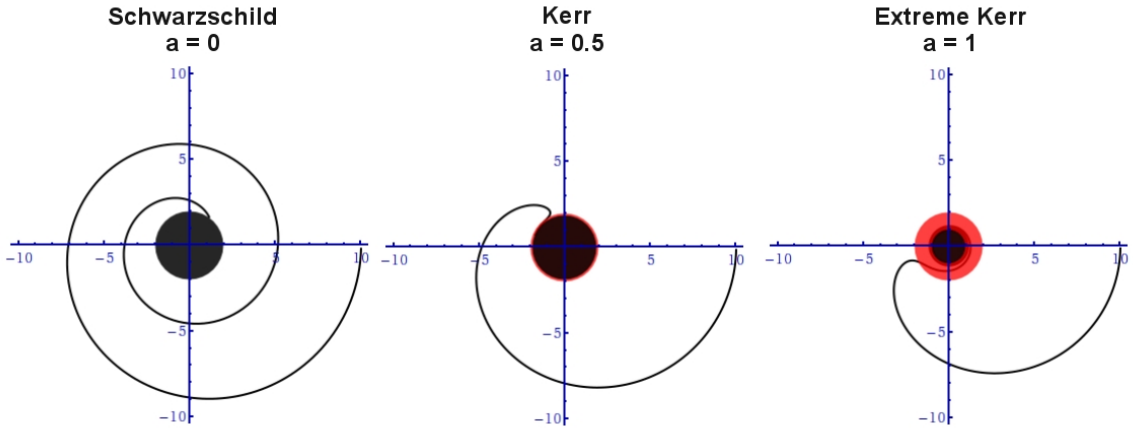

Figure 1.2: Various black hole inspiral orbits in the equatorial plane, with parameters $Jz = -10$, $e = 2.7$, and a starting radius of 10 meters. The central body has a mass of 1 m, while the spin parameter takes on values of 0, 0.5, and 1 m. At $a = 0$, the Schwarzschild case, the orbit spirals in with no decrease in coordinate angular velocity before impact. In the $a = 0.5$ case, the ergosphere is beginning to form (in red), which forces the orbits to reverse direction before impact. In the $a = 1$ case, the ergosphere is twice the diameter of the event horizon, and the incident particle reverses direction significantly before infall. Additionally, the particle wraps around the event horizon inside the ergosphere for several turns before falling in.

($\theta = 0$ and $\theta = \pi$), and bulging out at the equator. See Figure 1.1. This region between the event horizon and the second surface is known as the **ergosphere**.

The best way to physically understand the ergosphere is to look at the metric when $r_{eh} < r < r_{es}$. Anywhere in this region the timelike coordinate g_{00} has become a positive, spatial coordinate. However, the two off-diagonal terms are still negative, meaning that for any timelike geodesic in this region, the forward march of time is irrevocably linked to a positive $d\varphi$. It is physically impossible for any observer to maintain a stationary orbit within the ergosphere: for the sake of causality, they *must* rotate in the same direction as the black hole, regardless of their initial direction upon entering.

Of course, this rotation is not unique to the ergosphere. Even significantly far from the source, the effect should still be visible. This **frame dragging**, a fictitious force in the direction of rotation of the central object, is a natural byproduct of the spacetime in any rotating central body system. However, since the off-diagonal terms scale as $1/r$ in the far field regime, the effect is only obvious close to the central body. See figure 1.2.

Singularity

The physical singularity in Kerr is far less obvious than it's stationary Schwarzschild counterpart. As stated above, the proper way to distinguish between a physical spacetime singularity and a merely coordinate one is to examine the curvature scalar. Since the quantity $R^{\alpha\beta\gamma\delta}R_{\alpha\beta\gamma\delta}$ goes to infinity at a singularity *and* is invariant under coordinate transformation, we can use it to find physical singularities in Kerr. To that end, we write down the invariant in Boyer-Lindquist coordinates:[9]

$$R^{\alpha\beta\gamma\delta}R_{\alpha\beta\gamma\delta} = \frac{48m^2(r^2 - a^2\cos^2\theta)\left[(r^2 + a^2\cos^2\theta)^2 - 16r^2a^2\cos^2\theta\right]}{(r^2 + a^2\cos^2\theta)^6} \tag{1.14}$$

The denominator of the invariant is simply ρ^{12} from equation 1.10, implying the physical singularity at $r = 0$ and $\theta = \pi/2$. To better understand this result, we form $x^2 + y^2$ from equation 1.11, and are left with

$$\begin{aligned} x^2 + y^2 &= a^2 \\ z &= 0 \end{aligned}$$

Thus, we conclude that the Kerr singularity is a **ring singularity**, a circle of radius a in the x-y plane.

1.3 GRS 1915+105

It would certainly be possible to create an artificial system upon which to perform our analysis. We could create a central body of particular mass and spin, place a donor star at whatever orbital point we saw fit, and insert Gaussian variation into the system to generate the geodesics of less massive test particles streaming from one to the other. However, it seems nature has provided us already with an ideal toy system. The microquasar GRS 1915+105 is a binary system with a Kerr central body and a late main sequence donor star, from which gas is torn off by the black hole. The donor star has been identified via optical spectra to be of type K-M III (a cold red giant). This classification places a good restriction on both the mass and size of the star. That, combined with the period of the system and the inclination of the orbit, suggest a companion star mass of $1.2 \pm 0.2\ M_\odot$ and a radius of $21 \pm 4\ R_\odot$, or about 1.49×10^{10} m.[10] This in turn has suggested a central body mass of $14 \pm 4\ M_\odot$, and a lower bound on the spin parameter of $13.72\ M_\odot$ (98% of the mass).[11]

While spectral analysis has suggested the existence of an accretion disk for GRS 1915+105, there is very little experimental data regarding the size or mass of said disk. To fill in the blanks, we will employ a couple of results from classical orbital mechanics.

[9] Visser, p. 7.
[10] Greiner, 2001.
[11] McClintock et al., 2006

Table 1.1: Properties of Binary GRS 1915+105

	Values in SI	**Values in Solar Mass ($\mathbf{c = G = 1}$)**
Central Mass	2.78×10^{31} kg	$14\ M_\odot$
Companion Mass	2.39×10^{30} kg	$1.2\ M_\odot$
Orbital Seperation	7.54×10^{10} m	$5.11 \times 10^{7}\ M_\odot$
Disk Radius	2.48×10^{10} m	$1.68 \times 10^{7}\ M_\odot$
Stellar Radius	1.49×10^{10} m	$1.01 \times 10^{7}\ M_\odot$
Radial Velocity	1.49×10^{5} m/s	4.97×10^{-4}
Orbital Period	33.5 days	$5.88 \times 10^{11}\ M_\odot$
Central Body Spin	1.48×10^{4} Hz	$13.72\ M_\odot$

The first is the semi-major axis of the donar star, as described by Kepler's third law. Using the known period of 33.5 days, we find a semi-major axis of 7.53738×10^{10} m. The second result comes from the theory of Lagrange points. Classically speaking, the radius of an accretion disk is expected to start at the first possible circular orbit for any parcel of mass passing through the first Lagrange point (l_1). We simply quote the result here[12]

$$\begin{aligned} r_{disk} &= d\left[\frac{1}{2} - 0.227 \log_{10}\left(\frac{M_2}{M_1}\right)\right]^4 \left(1 + \frac{M^2}{M^1}\right) \\ &= 2.48 \times 10^{10} \text{ m} \end{aligned}$$

where d is the semi-major axis of the binary system. We consolidate the physical properties of the binary in Table 1.1

[12]Carroll and Ostlie, p. 666.

Chapter 2

Precession of the Perihelion

This chapter serves two functions. The first is to develop the numerical method that will be used for the remainder of this thesis. Using the substantially simpler case of a Schwarzschild black hole, we go through the process of developing the equations of motions from the Lagrangian and introducing the relevant constants. We then introduce the Runge-Kutta method, a numerical algorithm for solving first order ordinary differential equations. Finally, we show how to embed the Schwarzschild equations of motion within the Runge-Kutta algorithm, enabling us to calculate any geodesic so long as we know the energy, angular momentum, and starting position of the particle.

Once we have fully developed the method, we will then perform a simple test of the program using a well known example from the history of general relativity. One of the first phenomenological triumphs of relativity was Einstein's calculation of the anomalous precession of the perihelion of Mercury (figure 2.1). With the highest eccentricity and lowest orbital period of any of the planets in the solar system, Mercury's orbital precession has been a known problem since the mid-nineteenth century. While most of the precession could be explained via gravitational interactions with other planets or the oblateness of the sun, there was a remaining anomalous value of approximately 43 arcseconds per century which could not be explained by Newtonian physics. In 1915, Einstein was able to show that the additional precession was an inevitable result of general relativity. This would later be regarded as one of the first major successes of relativity theory. We will recreate here the same calculation, but we will do so by numerically calculating the geodesic directly, as opposed to the perturbative calculation of Einstein.

2.1 Equations of Motion

The standard method for describing motion in a curved spacetime is via Lagrangian mechanics. For the Schwarzschild geometry, we can write down the Lagrangian with ease:

$$\mathcal{L} = \frac{1}{2} g_{\mu\nu} \dot{x}^\mu \dot{x}^\nu = \frac{1}{2} \left[- \left(1 - \frac{2M}{r} \right) \dot{t}^2 + \frac{\dot{r}^2}{1 - \frac{2M}{r}} + r^2 \dot{\theta}^2 + r^2 \dot{\varphi}^2 \sin^2 \theta \right] \tag{2.1}$$

We apply the Euler-Lagrange equations (equation 5), and with a bit of rearranging are left with expressions for the second derivatives.

$$\ddot{t} = \frac{2M\dot{r}\dot{t}}{2Mr - r^2} \tag{2.2}$$

$$\ddot{r} = -\frac{2M\dot{r}^2}{2Mr - r^2} \tag{2.3}$$

$$\ddot{\theta} = -\frac{2\dot{r}\dot{\theta}}{r} \tag{2.4}$$

$$\ddot{\varphi} = -\frac{2\dot{\varphi}(\dot{r} + \dot{\theta}\cot\theta)}{r} \tag{2.5}$$

We now have the equations of motion in their familiar form, but for a test particle

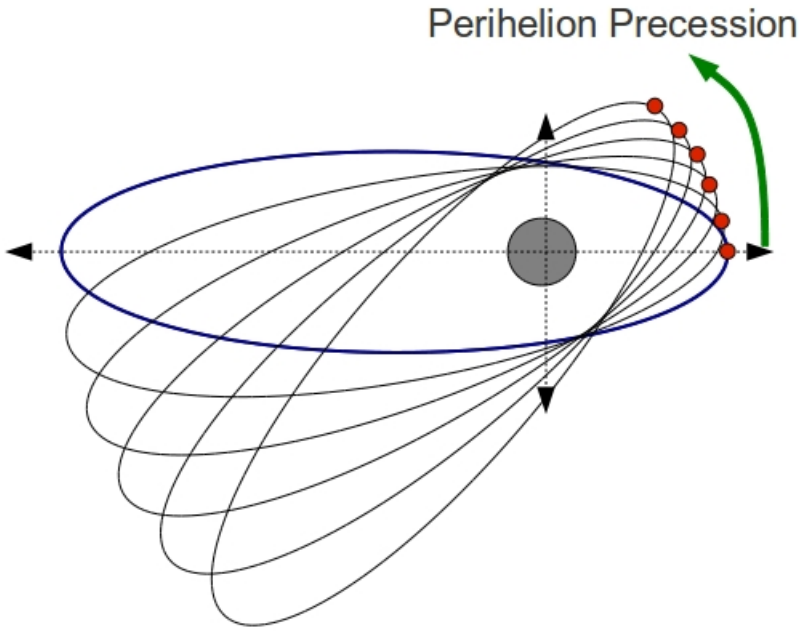

Figure 2.1: Perihelion precession of a elliptical orbit about a central body. The perihelion(s) are marked in red. In Newtonian theory (represented in blue), the point of closest approach would occur at the same polar angle at every pass; for a non $1/r$ potential, the orbits do not close back on themselves, and the perihelion slowly rotates after every pass (in green). We aim to calculate this precession for the case of Mercury.

in the Schwarzschild geometry. These are the same equations that we would have arrived at via the geodesic equation (6).

Note that equations 2.2-2.5 are four non-linear, coupled differential equations. Even with simplifying assumptions such as planar orbits ($\theta = \pi/2$ and $\dot{\theta} = 0$) we are left with three of the original four equations and minimal canceling cross-terms. Taking a cue from Newtonian orbital mechanics, our first inclination is to introduce constants of the motions, such as energy and angular momentum. This allows us to simplify the equations of motion significantly.

2.1.1 Constants of Motion

The first thing to notice about Schwarzschild motion is that any orbit is restricted to planar motion. Since a Schwarzschild black hole is spherically symmetric, we can rotate our coordinate axes such that any initial angular motion is only in the $\hat{z}$ direction. From equation 2.4, this implies $\dot{\theta} = \ddot{\theta} = 0$. Therefore, we should be able to describe any orbit in the $\theta = \pi/2$ plane using only three constants of the motion.

To that end, we recall the result from Lagrangian mechanics regarding constants of the motion known as Noether's Theorem: for any generalized coordinate that the Lagrangian does not explicitly depend on, the generalized momentum will remain constant. Notice that equation 2.1 does not depend on either φ or t (exactly as one would expect for a spherically symmetric stationary spacetime). This lets us write down

$$E = -\frac{\partial \mathcal{L}}{\partial \dot{t}} = \left(1 - \frac{2M}{r}\right)\dot{t} \tag{2.6}$$

$$L_z = \frac{\partial \mathcal{L}}{\partial \dot{\varphi}} = r^2 \dot{\varphi} \sin^2\theta = r^2 \dot{\varphi} \qquad \left(\text{when } \theta = \frac{\pi}{2}\right) \tag{2.7}$$

as constants of the motion.[1] We can also rotate any coordinate system such that the orbits lie in the equatorial plane. This gives us three of the four constants needed. As for the fourth, note that the Lagrangian as defined is just half the magnitude of the four-velocity. But since the four-velocity is always -1 for a timelike geodesic, the Lagrangian itself is a constant with value

$$\mathcal{L} = -\frac{1}{2} \tag{2.8}$$

We can now describe any orbit in the Schwarzschild geometry using these three constant parameters.

[1] These are called the energy and angular momentum from a careful comparison with the equivalent situation in Newtonian mechanics. The minus sign is by convention. For a more complete description, see Hartle, 2003 or Chandrasekhar, 1992.

2.2 The Method: Runge-Kutta

As previously stated, analytic solutions to equations 2.2-2.5 are difficult to come by in the best of circumstances. To describe generalized geodesic solutions, we must resort to numerical methods. What we are interested in is an algorithm that will solve ordinary differential equations. It should be able to solve the equations of motion to within a certain error tolerance once the full initial conditions have been specified. We need to be able to specify the initial orbital parameters (the starting four-position and four-velocity), and have it trace out that trajectory through the spacetime. We also need to employ a technique known as **adaptive stepping**: at each numerical point, the upper bound of the error is explicitly measured, and the size of the numerical step is adjusted to keep the error within a given tolerance. Although there are several such methods available, the **Runge-Kutta** algorithm is one of the most common and effective. It is simple, straight forward, and meets all the requirements we set. We will first describe the simplest form of the Runge-Kutta (RK) algorithm, then we will show how it works specifically for the orbital problem. Finally, we will describe how the adaptive stepping procedure allows us to save time on slowly changing parts of the motion while focusing in on the more interesting bits.[2]

2.2.1 Midpoint Runge-Kutta

The simplest type of RK procedure is the midpoint algorithm, so called because it essentially calculates the difference between two successive points, taking into account the difference in the function's value at the center point as well as the two end points. The main mathematical structure behind RK is the comparison between the Taylor expansion around a specific point with the unique features of the discrete derivative.[3]

Consider a first order linear differential equation of the form

$$\frac{df}{dx} = G(x, f) \tag{2.10}$$

that is, consider a function whose derivative is some function g of the original term f and the dependent variable x. We begin by Taylor expanding the left hand side around x to second order.

$$f(x + \Delta x) = f(x) + \Delta x \ f'(x) + \frac{1}{2}\Delta x^2 f''(x) + O(\Delta x^3) \tag{2.11}$$

Where the prime represents derivative with respect to x. To rewrite equation 2.11 in a more useful way, we take the second derivative from equation 2.10 and are left with

[2]This section relies heavily on Franklin, 2009 and Press et al., 1988. We use Franklin's notation for the Runge-Kutta Algorithm.

[3]That is, a derivative performed on discrete data, such that the Δx in the derivative becomes finite and small. IE

$$\frac{df}{dx} \approx \frac{f(x + \Delta x) - f(x)}{\Delta x} \tag{2.9}$$

$$\frac{dG}{dx} = f''(x) = G'(f,x) + \frac{dG}{df}f'(x)$$

We plug these in and (after a bit of suggestive rearrangement) are led to

$$\begin{aligned} f(x+\Delta x) &= f(x) + \Delta x\ G(f,x) + \frac{1}{2}\Delta x^2\left(G'(f,x) + \frac{dG}{df}f'(x)\right) + O(\Delta x^3) \\ &= f(x) + \Delta x\left[G(f,x) + \frac{1}{2}\Delta x\left(G'(f,x) + \frac{dG}{df}f'(x)\right)\right] + O(\Delta x^3) \end{aligned} \quad (2.12)$$

At this point a rather marvelous thing happens. If we look to the term in parenthesis, it strongly resembles the two-dimensional Taylor expansion of a function about the point x when evaluated at the point $x + \Delta x$. Explicitly

$$G\left(f + \frac{1}{2}\Delta x\ G(f,x)\ , x + \frac{1}{2}\Delta x\right) = G(f,x) + \frac{1}{2}\Delta x\left(G'(f,x) + \frac{dG}{df}f'(x)\right) + O(\Delta x^2) \quad (2.13)$$

Finally, we substitute 2.13 into 2.12 and we arrive at

$$f(x+\Delta x) = f(x) + \frac{1}{2}\Delta x\ G\left(f + \frac{1}{2}\Delta x\ G(f,x), x + \frac{1}{2}\Delta x\right) + O(\Delta x^3) \quad (2.14)$$

This is the equation for the midpoint Runge-Kutta method, a solution to any ordinary differential equation locally accurate to $O(\Delta x^3)$. The right hand side references only equations that we know from the start. Once we specify initial conditions x_0 and $f(x_0) = f_0$, we can systematically go through the range of interest in steps of Δx, determining the value of $f(x)$ at each point along the way.

The standard way to represent equation 2.14 is with a pair of intermediate terms, k_1 and k_2. Placing the equation on an evenly spaced grid of n points Δx apart, with $f_0 \equiv f(x_0)$, we can calculate all points on the grid via the following canonical algorithm

$$\begin{aligned} k_1 &= \Delta x\ G(f_j, x_j) \\ k_2 &= \Delta x\ G\left(f_j + \frac{1}{2}k_1,\ x_j + \frac{1}{2}\Delta x\right) \\ f_{j+1} &= f_j + k_2 \end{aligned} \quad (2.15)$$

This formulation is compact and involves completely sequential steps, making it ideal for numerical work. Furthermore, by including more terms in the Taylor expansion, we can increase the accuracy of the method to whatever level we want; however, each additional term brings with it an additional computation, significantly reducing the program run time. A common method which strikes a good balance between speed

and accuracy is the fourth order RK algorithm (RK4), which is accurate to $O(\Delta x^5)$. Note that each of the higher order RKs produces a different local numerical error level: midpoint gives $O(\Delta x^3)$, RK4 gives $O(\Delta x^5)$, RK5 gives $O(\Delta x^6)$, and so on. If we were to compare the numerical results of two different algorithms on the same function, say RK4 and RK5, we could produce an upper bound on the error at any given x_j. This is the trick that our adaptive stepping algorithm is based on, which we will come to in a moment.

2.2.2 RK Embedding of the Orbital Problem

Despite the immense cleverness of the Runge-Kutta method, we have so far only demonstrated its use on first order equations of a single variable. But equations 2.2-2.5 are four coupled equations of the second order in τ. We need to be able to embed the equations of motion in such a way as to allow us to use our RK algorithm. This turns out to be a surprisingly simple task.

Although we assumed scalar functions in the last section, there is nothing to stop us from rewriting equation 2.10 as a vector equation

$$\frac{d\vec{f}}{dx} = \vec{G}(\vec{f}, x) \tag{2.16}$$

Nothing is changed in the analysis, and we can just as easily solve for $\vec{f}$ as we could f. At first glance, this procedure seems a little suspect. It is claiming that *any* coupled vector equations can be solved by taking steps in Δx, no matter how many complicated cross terms are present in $\vec{G}$. But look at the right hand side of equation 2.15: nowhere is the f_{j+1} present, which means that as long as we calculate each component of the vector at every step, the process works perfectly. In the case of vector equations, this works *as long as we can decouple the equations* into the form of equation 2.16.

The second discrepancy is the fact that our equations of motions are of second order form, whereas RK only works for first order. This is solved easily enough, since any second order equation can be reduced to first order. Look to the example

$$f''(x) = a(x)f'(x) + b(x)f(x) + c(x) \tag{2.17}$$

If we introduce an auxiliary function, $z(x) \equiv f'(x)$, we can write equation 2.17 as

$$\begin{pmatrix} f'(x) \\ z'(x) \end{pmatrix} = \begin{pmatrix} z(x) \\ a(x)z(x) + b(x)f(x) + c(x) \end{pmatrix} \tag{2.18}$$

and voila, a second order equation reduced to first order.

With these two tricks in hand, it is time to embed our equations of motion in a form appropriate to Runge-Kutta. Using the velocities as the intermediate functions (the $z(x)$ function above), we have

$$\frac{d}{d\tau}\begin{pmatrix} t \\ \dot{t} \\ r \\ \dot{r} \\ \theta \\ \dot{\theta} \\ \varphi \\ \dot{\varphi} \end{pmatrix} = \begin{pmatrix} \dot{t} \\ \ddot{t} \\ \dot{r} \\ \ddot{r} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\varphi} \\ \ddot{\varphi} \end{pmatrix} = \begin{pmatrix} \dot{t} \\ \frac{2M\dot{r}\dot{t}}{2Mr-r^2} \\ \dot{r} \\ -\frac{2M\dot{r}^2}{2Mr-r^2} \\ \dot{\theta} \\ -\frac{2\dot{r}\dot{\theta}}{r} \\ \dot{\varphi} \\ -\frac{2\dot{\varphi}(\dot{r}+\dot{\theta}\cot\theta)}{r} \end{pmatrix} \tag{2.19}$$

We can also write this in a form more appropriate to numerical work, in terms of equation 2.15

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \end{pmatrix}_{j+1} = \begin{pmatrix} f_2 \\ \frac{2Mf_2f_4}{2mf_3-(f_3)^2} \\ f_4 \\ -\frac{2M(f_4)^2}{2mf_3-(f_3)^2} \\ f_6 \\ -\frac{2f_4f_6}{f_3} \\ f_8 \\ -\frac{2f_8(f_4-f_6\cot f_5)}{f_3} \end{pmatrix}_{j} \tag{2.20}$$

We can now numerically calculate any geodesic in the Schwarzschild geometry, provided we can specify the proper initial conditions.

2.2.3 Adaptive Stepping

Although the RK4 algorithm is quite good for most applications, there are particular advantages to using an adaptive stepping technique. Take for example the case of elliptical orbital motion.

For the regions with a higher acceleration (the perihelion and aphelion, in this case), the geodesic changes more rapidly per unit τ than during the middle of the orbit. The RK4 method samples at equal intervals in $\Delta\tau$. If we pick a $\Delta\tau$ that is adequate for sampling the semi-minor regions of the orbit, then the faster regions at the ends would be insufficiently covered by the grid to give a decent picture of the orbit. In particular, if we are looking to extract information about the precession, then we need a very dense sampling of points around the perihelion. On the other hand, if we decrease $\Delta\tau$ sufficiently so as to give us information about the perihelion and aphelion, then the semi-minor axes are vastly over sampled, wasting both computation time and memory resources.

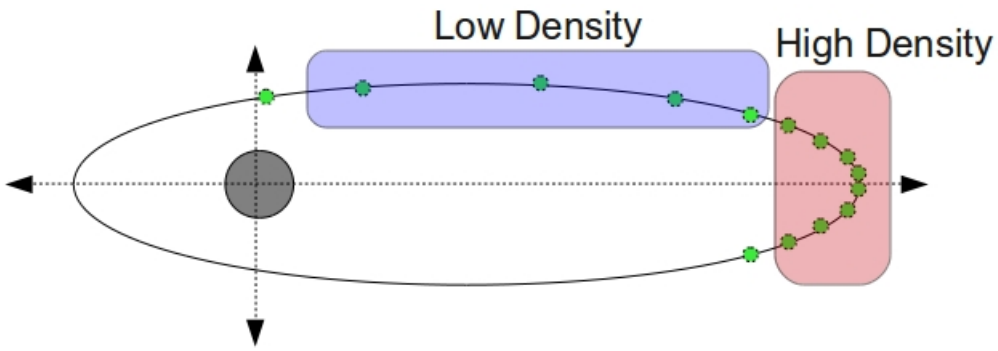

Figure 2.2: Example of adaptive stepping around a elliptical orbit. The green dots represent the grid sampling points τ_j. The adaptive stepping routine increases the sampling size at the aphelion (in red), and decreases it in less interesting regions (in blue).

What we want instead is a way to dynamically adjust $\Delta\tau$ so that the perihelion is in a region of high sampling, and the semi-minor axes are sampled more broadly. The **54 Runge-Kutta** algorithm allows us to do this. By calculating the results of both the RK5 and RK4 methods, we can obtain an upper bound on $\Delta\tau$ by comparing the results. Rather than explicitly calculating both, we can actually just calculate the RK5 results, then use the difference in the k_i coefficients to determine the error. That is, when we calculate a fifth order RK of the form

$$f_{n+1}^{(5)} = f_n^{(5)} + c_1^{(5)}k_1 + c_2^{(5)}k_2 + c_3^{(5)}k_3 + c_4^{(5)}k_4 + c_5^{(5)}k_5 + c_6^{(5)}k_6 + O(\Delta\tau^6)$$

we can also calculate a fourth order RK embedded in a fifth order equation by manipulating the coefficients such that

$$f_{n+1}^{(4)} = f_n^{(5)} + c_1^{(4)}k_1 + c_2^{(4)}k_2 + c_3^{(4)}k_3 + c_4^{(4)}k_4 + c_5^{(4)}k_5 + c_6^{(4)}k_6 + O(\Delta\tau^5)$$

where the superscripts represent the order of the method.[4] Since the magnitude of the error is known in both cases, we can form an upper limit on the error of any individual step by taking the difference of the two values.

$$\epsilon_{actual} = f_{n+1}^{(5)} - f_{n+1}^{(4)} = \sum_{i=1}^{6}(c_i^{(5)} - c_i^{(4)})k_i \tag{2.21}$$

Note that equation 2.21 is actually independent of the final value of the RK4 evaluation. With the numerical values of $c_i^{(5)} - c_i^{(4)}$, we can estimate the error to

[4]The RK4 runs from the RK5 starting point, since we are only interested in the error between individual steps, and not the method over the entire domain. Therefore, when we talk about an error bound, we mean a *local* bound between individual steps, not an overall error in the geodesic.

accuracy $\Delta\tau^5$ at each step of the RK5 algorithm.[5] We are then able to determine what step size *would* have been necessary to yield a given desired accuracy. The error goes as the fifth power of the step size $\Delta\tau$, allowing us to write down the modified step size as

$$\Delta\tau_{needed} = \Delta\tau \left| \frac{\epsilon_{needed}}{\epsilon_{actual}} \right|^{0.2} \tag{2.22}$$

With a specified error tolerance, ϵ_{needed}, equation 2.22 actually serves two functions: when the error produced by the initial step $\Delta\tau$ is greater than the specified tolerance, we have $\epsilon_{needed} < \epsilon_{actual}$. We can use 2.22 to tells us what step we would have needed to take to stay within error. We can then perform the calculation again, with step size $\Delta\tau_{needed}$, ensuring that our error is properly bounded. On the other hand, if the error produced by the initial step is smaller than our error tolerance, then we are wasting computational resources by using a smaller $\Delta\tau$ than needed to properly sample the space. In that case, 2.22 tell us how much we can increase $\Delta\tau$ for the next step, saving time and resources. Because of this, the 54 Runge-Kutta is substantially more efficient and more accurate than either the RK5 or RK4 methods, and is the method we use here.

There is one point in the error calculation that we have neglected. Notice that the numerical error of equation 2.21 is actually a vector, whereas the step size of equation 2.22 is always a scalar quantity. We need some way to compute a scalar magnitude of the vector error if we want to use our 54 Runge-Kutta. Although technically any method of computing the norm will do, the language of general relativity has already provided us with a natural scalar invariant in the form of the line element. If we assume the error between the RK5 and RK4 calculations to be small, then we can replace the differentials in the line element with the corresponding error vector terms; that is

$$ds^2 = g_{\mu\nu}(dx^\mu)(dx^\nu) \rightarrow E_{actual} = g_{\mu\nu}\epsilon^\mu\epsilon^\nu \tag{2.23}$$

If we evaluate the metric at the midpoint between the RK5 and RK4 predictions, then equation 2.23 gives us a physical way to describe the error at any given point. Of course, we need to calculate the error for the position components, x^μ as well as the velocity components $\dot{x}^\mu$. Since we need both norms for the error, we take the Cartesian norm of the two errors, and minimize that quantity. While this sacrifices some of the physical interpretation, we still have an error corrector based in the physical geometry of the system.

2.3 Precession of the Perihelion

Using the algorithms discussed in the previous section, we constructed an adaptive stepping RK54 integrator to generate geodesics in the Schwarzschild geometry. After

[5] The values for the c_i's and the k_i's we use here are the commonly used Cash-Karp coefficients. See Press et al., 1988, p. 717.

determining the correct inputs for the case of Mercury, we numerically calculate the precession of the perihelion, and compare our result to the known experimental value. Although the analysis for this chapter has been based on the Schwarzschild spacetime, the equations used in the program were those for the Kerr geometry, but since the Kerr spacetime reduces to Schwarzschild in the static limit (when $a = 0$), the end result is exactly the same.[6]

2.3.1 Orbital Parameters of Mercury

To determine the orbital parameters for the simulation, we first look at what observational data we have. We know the perihelion and aphelion of the system occur at 4.6001×10^{10}m and 6.9817×10^{10}m respectively.[7] While we could use experimental data to determine the other input parameters (with the exception of $\dot{t}$), there is a much simpler way. Recall that the Schwarzschild Lagrangian is itself a constant of the motion (equation 2.8). The simulation was run in units of solar mass, so $M = 1$, and we know that the orbit of Mercury stays in the $\theta = \pi/2$ plane. If we also assume that $\dot{r} = 0$ at the perihelion and aphelion, we can simplify the Lagrangian (equation 2.1) to

$$-1 = -\left(1 - \frac{2}{r_{tp}}\right)\dot{t}^2 + r_{tp}^2\dot{\varphi}^2$$

Where r_{tp} is the distance to the sun at either of the turning points in the orbit. Since we are interested in the parameters in terms of the energy and angular momentum of the system, we can insert those expressions (equations 2.6 and 2.7) into the Lagrangian to get

$$-1 = \frac{E^2\left(2r_{tp} - r_{tp}^2\right)}{(r_{tp} - 2)^2} + \frac{L_z^2}{r_{tp}^2} \tag{2.24}$$

With two known values of r_{tp}, we have two equations in two unknowns, and can solve for the energy and angular momentum, yielding

$$E = 0.999999987\ M_\odot$$
$$L_z = 6.136.0634\ M_\odot^2$$

Using these initial conditions, we simulated the orbit of Mercury over a hundred year span (about $6.4 \times 10^{14}\ M_\odot$). To extract the perihelion at each point, a minimum finder was implemented about the radial component. It would record the φ term at each closest approach of Mercury, recording exactly where each point occurred. The

[6] The only notable difference is a fourth constant of the motion that appears in Kerr spacetime. This will be discussed in the next chapter, but for Schwarzschild orbits in the equatorial plane, the Carter constant reduces to $K = {L_z}^2$ anyway, making it a non-issue.

[7] Bakick, 2000, p. 84

results are shown on the next two pages, in figures 2.3 and 2.4. The simulation was performed at different error tolerances of 10^{-3}, 10^{-5}, 10^{-7}, and 10^{-9} "meters". In each case, the precession rate oscillates about the actual value, with the variation decreasing significantly with the error tolerance. At the final error level of a billionth of a meter, the numerical results are barely distinguishable from the analytic and measured value. Quantitatively, the numerical result achieves a slope of 42.95 ± 0.11 arcseconds per century, with an R^2 value of 0.997. This value is within %0.09 of the accepted value of 42.98 arcseconds per century.[8]

[8]Taylor and Wheeler, 2000, p. C-11.

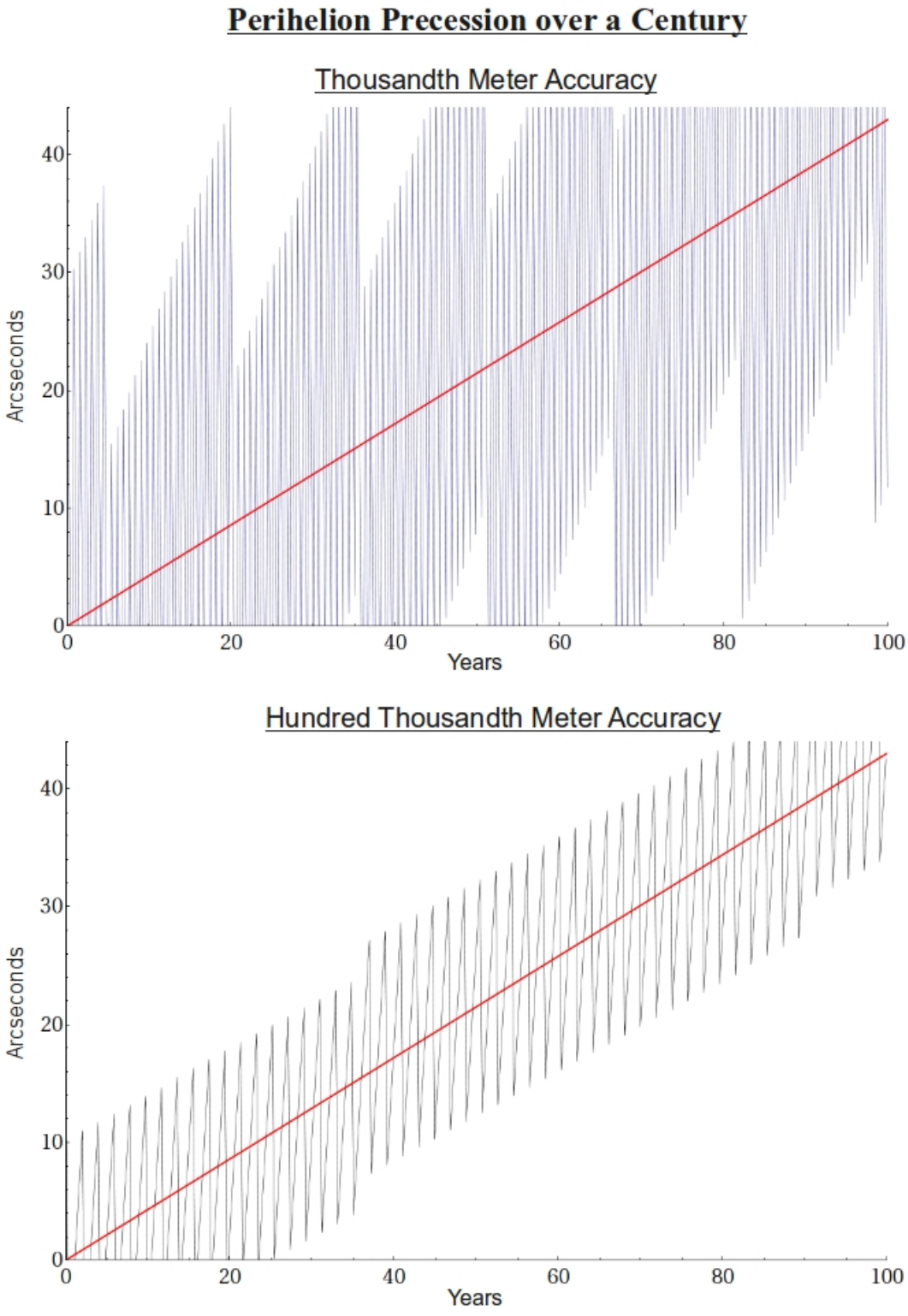

Figure 2.3: Precession of the perihelion of Mercury, numerically measured (in blue) at error tolerances of a thousandth and hundred thousandth meters. The red line represents the known value, crossing the 43 arcsecond mark after 100 years.

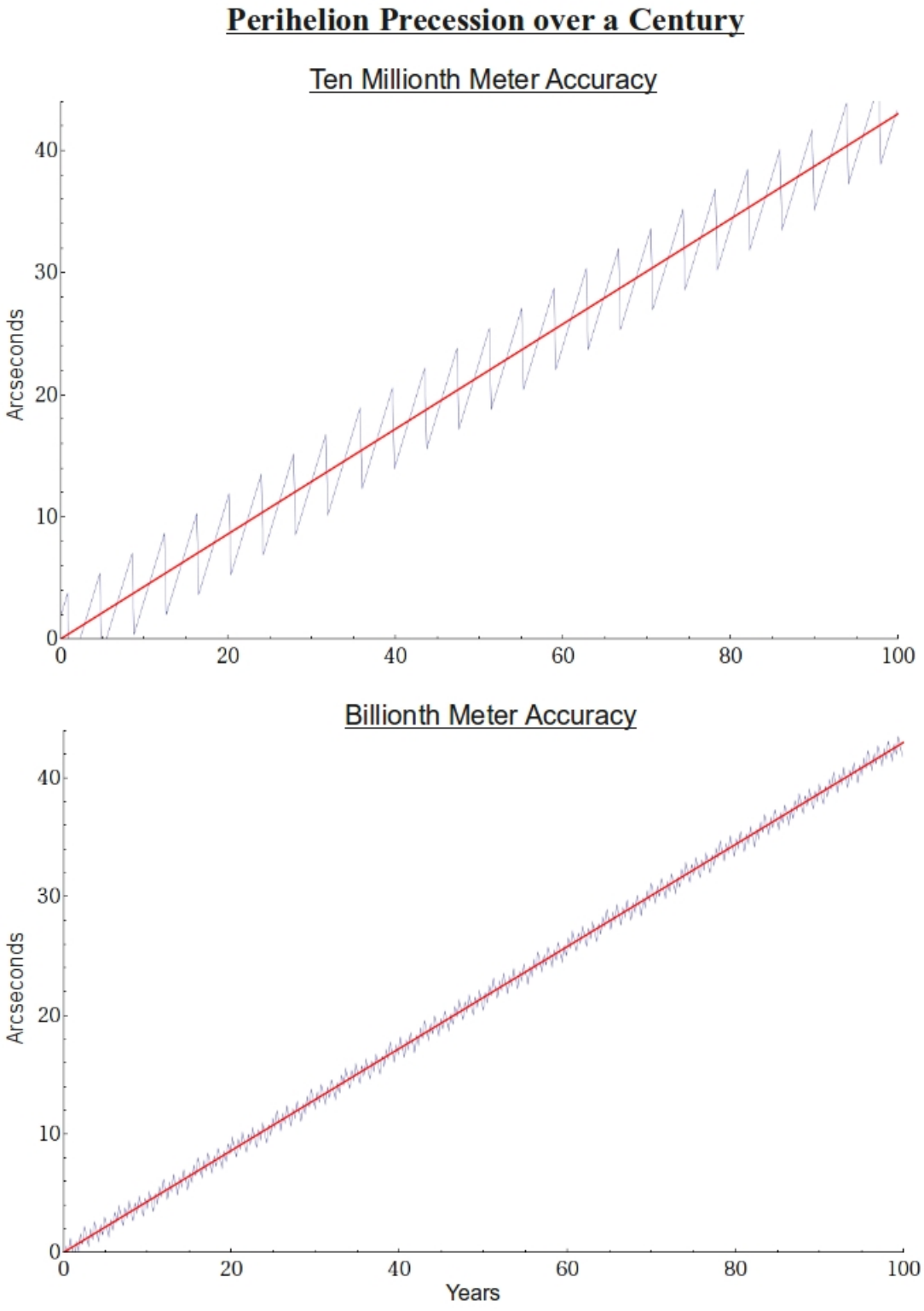

Figure 2.4: Precession of the perihelion of Mercury, numerically measured (in blue) at error tolerances of a ten millionth and billionth meters. The red line represents the known value, crossing the 43 arcsecond mark after 100 years.

Chapter 3

Extreme Kerr Accretion Disks

In this chapter, we describe some of possible geodesics that exist in a binary system with a rotating central body. What we are particularly interested in are mass transfer geodesics, those that describe a particle's path as it falls off a companion star, entering the disk of gas surrounding the black hole, and eventually falling through the event horizon. We achieve this by including a damping term in the Kerr equations of motion, representing drag forces within the gas disk.

Rather than an intensive analysis of the group properties of these accretion disk geodesics, we will focus on the qualitative difference between two simulations of GRS 1915+105; one with the spin of the black hole **parallel** to the spin of the incident particles, and one with the black hole spin opposite (or **anti-parallel**) to that of the incident particles. Recall from Chapter 1 that particles experience a fictitious force whenever they are in the presence of a rotating central mass. This frame dragging is expected to sap and eventually counter the incident angular velocity of the particles traveling against it (the anti-parallel case). This effect, applied to a large quantity of geodesics comprising an accretion disk, is what we hope to observe.

3.1 Kerr Equations of Motion

We obtain the equations of motion for the Kerr geometry in identical fashion to the previous section, this time using the Kerr Lagrangian

$$\mathcal{L} = \frac{1}{2}\Bigg[-\left(1 - \frac{2Mr}{\rho^2}\right)\dot{t}^2 + \frac{\rho^2}{\Delta}\dot{r}^2 + \rho^2\ \dot{\theta}^2 + \left(r^2 + a^2 + \frac{2a^2 Mr\sin^2\theta}{\rho^2}\right)\sin^2\theta\ \dot{\varphi}^2 - \left(\frac{4aMr\sin^2\theta}{\rho^2}\right)\dot{t}\ \dot{\varphi}\Bigg] \tag{3.1}$$

where ρ^2 and Δ are defined in equation 1.10. Applying the Euler-Lagrange equation to this gives us the equations of motion for Kerr geodesics. The equations themselves are highly non-linear, coupled, and unwieldy; they are reproduced elsewhere. They are also far too complicated to be solved analytically. Although certain orbits can be

derived using various simplifying assumptions,[1] for all other cases we must turn to numerical solutions. We can use our Runge-Kutta integrator from before, albeit with new constants of motion for the Kerr geometry.

3.1.1 Kerr Constants of the Motion

As with the Schwarzschild case, we can use the energy and angular momentum as constants of the motion. In Kerr, they become

$$E = -\frac{\partial \mathcal{L}}{\partial \dot{t}} = \left(\frac{2aMr\sin^2\theta}{\rho^2}\right)\dot{\varphi} + \left(1 - \frac{2Mr}{\rho^2}\right)\dot{t} \tag{3.2}$$

$$L_z = \frac{\partial \mathcal{L}}{\partial \dot{\varphi}} = \left(\frac{\left((a^2+r^2)^2 - a^2\Delta\sin^2\theta\right)\sin^2\theta}{\rho^2}\right)\dot{\varphi} - \left(\frac{2aMr\sin^2\theta}{\rho^2}\right)\dot{t} \tag{3.3}$$

Note that in the stationary limit, $a \to 0$, $\rho \to r$, and these exactly match the constants from the Schwarzschild case (equations 2.6 and 2.7).

If this were the Schwarzschild geometry, we would be done at this point; we could rotate our coordinate system until any orbit lay in the $\theta = \pi/2$ plane with $\dot{\theta} = 0$, and compute the geodesics accordingly. Unfortunately, the Kerr metric is *not* spherically symmetric. When we introduce spin to the central body, we also introduce a preferred direction into the spacetime. This axial symmetry gives us free range over the φ component of the system, but we can no longer perform rotations in the θ coordinate. To do so would change the direction of the effective frame dragging forces in the φ direction.

Instead, we introduce a new constant of the motion. The **Carter constant** is derived directly from the symmetry of the spacetime[2]. In terms of the metric coordinates, it is given by[3]

$$K = \frac{1}{\Delta}\left(\Delta\dot{t} - a\ \Delta\ \dot{\varphi}\sin^2\theta\right)^2 - \frac{\rho^4}{\Delta}\ \dot{r}^2 + r^2 \tag{3.4}$$

This formulation of the Carter constant allows us to form K if we know the original coordinate properties of the system. Equation 3.4 can also be written in terms of E and L_z as[4]

$$K = (aE\sin\theta - L_z\csc\theta)^2 + \rho^4\dot{\theta}^2 - a^2\cos^2\theta \tag{3.5}$$

Looking at the stationary case, we note that as $a \to 0$ and $\theta \to \pi/2$, the Carter constant goes as $K \to L_z^2$, and is no longer a unique constant of the motion.

[1] Such as planar orbits, specific ratios of energy and angular momentum, linearized Kerr (weak field), etc. See Chandrasekhar, 1992.

[2] Specifically, it is a quadratic conserved quantity, in the form of a second rank Killing tensor. Contrast this with the linear Killing vectors for the t and φ coordinates that produce E and L_Z. See Chandrasekhar, 1992, p. 343.

[3] Equation 3.4 applies *only* for timelike geodesics. For null geodesics, the final r^2 term vanishes.

[4] For null geodesics the $a^2\cos^2\theta$ term vanishes.

3.2 Orbital Parameters for GRS 1915+105

From Chapter 1, we know some of the orbital parameters for the simulation we wish to create. To convert the observed parameters into usable starting conditions for our simulation, we use the orbital separation, and proceed to determine the parameters for the companion star. We cannot just set $\dot{r} = 0$ as we did for the Schwarzschild case: for the circular case, $\dot{r} = 0$ at *every* point, not just the perihelion and aphelion. To get our two equations in two unknowns, we must also use $\ddot{r} = 0$.

First, we must reparameterize the Lagrangian in terms of the Kerr energy and angular momentum. If we assume that the companion star lies in the equatorial plane, we can insert equations 3.2 and 3.3 into equation equation 3.1 to get

$$\mathcal{L} = \frac{r^3(\dot{r}^2 - E^2) + r(L_z^2 - a^2E^2) - 2M(L_z - aE)^2}{2r\Delta} \tag{3.6}$$

Because the Lagrangian itself is a constant of the motion, we can set equation 3.6 equal to $-1/2$, and solve for $\dot{r}^2$

$$\dot{r}^2 = \frac{(E^2-1)r^3 + 2Mr^2 + \big(a^2(E^2-1) - L_z^2\big)r + 2M(L_z - aE)^2}{r^3} \tag{3.7}$$

which gives us our expression for $\dot{r}$. To get $\ddot{r}$, we differentiate $\dot{r}^2$ and solve for the second derivative, yielding

$$\ddot{r} = \left(\frac{d}{d\tau}\dot{r}^2\right)\frac{1}{2\dot{r}}$$

Finally, we can replace the term in parentheses with the time derivative of equation 3.7 and we obtain

$$\ddot{r} = \frac{-Mr^2 + \big(L_z^2 - a^2(E^2-1)\big)r - 3M(L_z - aE)^2}{r^4} \tag{3.8}$$

With this, we have two equations in two unknowns, and can solve for the energy and angular momentum of any circular orbit in the equatorial plane once we know r, M, and a. Using the values from Table 1 and setting equations 3.7 and 3.8 equal to zero, we find the energy, angular momentum, and Carter constant (using equation 3.5) of the donor star as

$$\begin{aligned} E &= 0.99999986\ M_\odot \\ L_z &= 26729.08882\ M_\odot^2 \\ K &= 7.13711 \times 10^8\ M_\odot^2 \end{aligned}$$

Inserting these values into our program gives a nearly circular orbit, precisely as we would expect.

With the central body simulated, we have a starting point for our accretion disk simulation. If we want to simulate randomly placed particles about the companion star, the natural first step is to pick initial conditions from a Gaussian distribution

in parameter space. The mass and spin of the central body will remain unchanged, but we want to vary the initial starting positions in r, θ, and φ, as well as the initial velocity, in the form of E, L_z, and K. We will begin with the position distribution.

For the particle starting positions, we actually want a three dimensional Gaussian, centered on the middle of the companion star with the majority of particles starting within the outer radius of the star itself. To that end, we define the Gaussian mean to be the orbital parameters derived for the donor star above. We then determine, using the same procedure as above, what the energy, angular momentum, and Carter constant will be for the circular orbits located at the innermost and outermost points on the star (the orbital separation plus and minus the stellar radius). We then set the 2σ point of the Gaussian to correspond with these ranges, explicitly quantifying our assumption that the majority of particles start from within the star.[5] For the position components (r, θ, φ), the ranges are geometrically found to be:

$$[r] = 1.98 \times 10^7 M_\odot$$
$$[\theta] = [\varphi] = 0.383 \text{ Rad}$$

To calculate the range of energy and angular momenta, we calculate E and L_z for circular orbits at the innermost and outermost points on the star. The difference between those two values is assumed to be the range of the constants. Running through this procedure yields

$$[E] = 5.5 \times 10^{-8}\ M_\odot$$
$$[L_z] = 5204.321\ M_\odot^2$$

Since we also expect there to be very little angular motion in directions outside the equatorial plane, we can compute the variation in the Carter constant by calculating the value at the highest point out of the plane on the stellar surface. We use the energy and angular momentum computed for the central body for this calculation, since our analysis in equations 3.6 through 3.8 depended on the assumption that $\theta = \pi/2$, and redoing the procedure for positions outside of the plane is exceedingly difficult. With this, we use equation 3.5, and obtain the range of Carter constants for our system

$$[K] = 2.683 \times 10^7\ M_\odot^2$$

With one exception, we now have the Gaussian parameters for our simulation. The energy calculated here only represents the gravitational energy between the points of closest and farthest approach in the star. Since any gas particles produced within the stellar environment will have kinetic energies and velocities significantly different in

[5] With 2σ, we are assuming that about 94.45% of the particles originate within the star. We chose a relatively large standard deviation to allow for the fact that much of the gas comes from the stellar surface.

Table 3.1: Gaussian Parameters for Accretion Disk Geodesics

Parameter	Mean	Range	Standard Deviation
r ($M_\odot$)	5.103×10^7	1.978×10^7	4.945×10^6
θ (rad)	$\pi/2$	0.383	0.096
φ (rad)	0	0.383	0.096
E ($M_\odot$)	0.99999986	5.5×10^{-8}	1
$\mathbf{L_z}$ ($M_\odot^2$)	26729.089	5204.321	N/A
K ($M_\odot^2$)	7.137×10^8	2.683×10^7	7.158×10^6

range than the circular orbit, we let the standard deviation of the energy range over the *entire* range of energies (that is, $\sigma_E \approx 1$)

These results are consolidated in Table 3.1. Although the gas particles are significantly less massive than the companion star, the geodesic equation does not take this into account. In Newtonian dynamics, this difference would be very obvious in the angular momenta of the particles. But with Kerr angular momentum (equation 3.3), this distinction is nowhere to be found. However, we do know that the angular velocity of the particles, $\dot{\varphi}$, should be roughly equal to the angular velocity of the star. If we take the stellar value for $\dot{\varphi}$, and combine it with the energy and position of the particles chosen from the Gaussian distribution, we can compute a more reasonable angular momentum by combining equations 3.2 and 3.3 to yield

$$L_z = \frac{\sin^2\theta\,(\rho^2\Delta\dot{\varphi} - 2MaEr)}{a^2\cos^2\theta - 2Mr + r^2} \tag{3.9}$$

With this expression for angular momentum, we now have a distribution from which to pick initial conditions for our accretion disk geodesics.

3.3 Disk Damping

Unfortunately, it is not enough to include Gaussian variation in our input parameters. Most of the geodesics we generate will still be stable or semi-stable elliptical orbits which do not fall into the central black hole. In astrophysical accretion, it is expected that the majority of energy loss contributing to the infall of gas comes from thermal effects; the drag and internal viscosity of the disk causes energy to be radiated away from the gas from thermal emission.[6] Since we are not simulating any interaction between the particles, we need some way to simulate that drag force.

The actual interactions between these particles and the hydrodynamics of mass transfer gas is currently very poorly understood. Rather than attempt to make a dynamic simulation based on current theories, we decided to use a simple damping term, proportional to the three velocity, and inversely proportional to the distance from the central body.

[6] Carroll and Ostlie, 2007, p. 661.

$$(F_{damp})^a = -\frac{b}{r}\,\dot{x}^a\,\Theta(r - r_{disk}) \tag{3.10}$$

where b is a constant, r is the radial component of the four-vector, and Θ is the Heaviside theta function. The vector index runs only over the spatial component.

The rationale behind this equation is rooted in a basic mechanics problem. The $-b\,\dot{x}^a$ term is simply the velocity dependent drag force from elementary mechanics, representing a frictional force. The b coefficient was determined phenomenologically by adjusting and fine tuning the order of magnitude until the majority of particles incident on the accretion disk would remain in the disk and eventually spiral into the black hole. The Heaviside theta term serves as a boundary condition: if we assume that the disk starts at the classical disk boundary calculated in Chapter 1 ($r_{disk} = 1.68 \times 10^7\ M_\odot$), then the Heaviside will turn on the damping only when the particle enters the disk, exactly as we want.[7]

3.4 Proper Time Considerations

Up until this point, we have talked about the simulation from the point of view of the particle itself. Far from the source, the distinction between proper time and coordinate time is tiny.[8] For this reason, we have been justified in treating every proper time particle as a particle observed in coordinate time. But as we move on to strong field geodesics, specifically the infall geodesics that we are interested in for the accretion disk, we must make this distinction clear.

First is the problem of the simulation itself. As written in Chapter 2, our Runge-Kutta program calculates geodesics in proper time. This is fine if we are only looking to a single particle at once. But our aim is to compare the geodesics from several thousand particles simultaneously. It is not at all clear how to compare thousands of particles at different points in spacetime with different proper times. Instead, we opt to transform to coordinate time for all simulated geodesics. This describes what an observer far from the source (such as an astronomer on Earth) would see. In terms of our simulation, this task is actually quite simple.

Recall from Chapter 2 (equations 2.19 and 2.20) that our Runge-Kutta embedding for the orbital problem takes the general form

$$\frac{d}{d\tau}\vec{f} = \vec{G}(\vec{f}, \vec{x}) \tag{3.11}$$

[7]Since our numerical method relies on a smooth geodesic for its computation, having a sudden discontinuity when the particle crosses the disk boundary must be avoided. To that end, we used an analytic approximation to the Heaviside theta:

$$\Theta(r - r_{disk}) \equiv \frac{1}{1 + e^{100(r - r_{disk})}}$$

[8]e.g. for every 1 second that passes in coordinate time, a proper time observer on the donor star for GRS 1915+105 would experience 1.0000004 seconds.

To transform to coordinate time, we take a cue from elementary calculus, and simply multiply both sides by $\frac{d\tau}{dt}$

$$\left(\frac{d\tau}{dt}\right)\frac{d}{d\tau}\vec{f} = \left(\frac{d\tau}{dt}\right)\vec{G}(\vec{f},\vec{x})$$
$$\frac{d}{dt}\vec{f} = \left(\frac{d\tau}{dt}\right)\vec{G}(\vec{f},\vec{x})$$

Now note that $\frac{d\tau}{dt}$ is just the reciprocal of the first component of the proper velocity, $\dot{t}$. But $\dot{t}$ is just the second component of our Runge-Kutta embedding for the orbital problem! This lets us rewrite equation 3.11 as

$$\begin{aligned}\frac{d}{dt}\vec{f} &= \left(\frac{1}{\dot{t}}\right)\vec{G}(\vec{f},\vec{x}) \\ &= \frac{\vec{G}(\vec{f},\vec{x})}{f^2}\end{aligned} \tag{3.12}$$

where f^2 is the second component of $\vec{f}$. By dividing the equations of motion by $\dot{t}$, we can express any geodesic calculated in proper time as an observed geodesic in coordinate time.

In addition to running in coordinate time, the simulation was written to output two sets of data. The first set recorded the positions and velocities at every step of the Runge-Kutta algorithm. This is perfectly suitable for analyzing single geodesics, but since the adaptive stepping of the program recorded data at "random" points at time (stemming from the adaptive stepping in Δt), it is somewhat difficult to compare the output of two runs. Therefore, a second output file was generated, which recorded the same information, but at equal intervals in coordinate time. This allows us to plot and analyze multiple geodesics simultaneously, since the stepping Δt between sampling points is identical. This procedure does reintroduce the problem mentioned in Chapter 2, where large quantities of unnecessary data is produced in slow moving regions, and the geodesic is under-sampled in fast moving regions. Despite this, we are still sampling from an error bounded geodesic, and can go back and check the cleanly sampled data if need be.

Finally, there is the issue of the geodesics crossing the event horizon. In proper time, an observer falling into the black hole experiences no time dilation, and will appear to fall straight through the event horizon without impediment. But in coordinate time, the time dilation factor increases as the geodesic approaches the event horizon, and becomes infinite as the particle crosses the horizon. Practically speaking, this means that any simulated particle which falls into the black hole will appear to stick to the event horizon forever. Since there is no point in calculating the trajectory for an object that will remain fixed until the end of time, we engineered an "event horizon tolerance" into our simulation. Once the particle crossed the tolerance boundary (1/100 $M_\odot$ from the horizon by default), the program would report the particle as

captured by the black hole, and discontinue the simulation.[9]

3.5 Accretion Disk Simulation

With the procedure and starting conditions described in the previous section, we proceeded to run the simulation 10,000 times for both the parallel and the anti-parallel cases, with random staring parameters selected from the Gaussian distributions.[10] Each simulation was run with a coordinate time spacing of $3 \times 10^5\ M_\odot$ (or about 3/2 seconds), for a duration of $2 \times 10^{12}\ M_\odot$ (1/3 years, or about 3.4 orbital periods of the system). This was far more time than was needed for the accretion geodesics to spiral in, but since the simulation terminated upon particle capture, there was no wasted computational time.

While perturbing a circular orbit, it was inevitable that some of the particles would actually follow a larger orbit than the donor star, and would never cross the damping boundary. Since we were only interested in those geodesics which would conspire to create an accretion disk, we removed the particles which never crossed r_{disk}. Of the 10,000 simulations that were run, approximately 40% were found to be disk geodesics. These ∼4,000 are the ones we restrict our attention to.

Figure 3.1 shows a small sample of the output data produced. Shown in red are thirty of the accretion disk geodesics that were randomly produced by the simulation. The behavior is exactly as we hoped: after leaving the companion star, the gas particles float towards the black hole in an elliptical orbit. Once they hit the damping region in gray, they lose energy quickly, and spiral into the singularity. Approaching the central body, the density increases rapidly, forming a thin disk that extends out of the equatorial plane.

Drawing the geodesics in this way is good from a visualization point of view, but limited in the number of particles that can be shown. What we want is a way to see a cross section of all the geodesics. That is, a program that will keep track of every point at which a particle passes through the plane, and record each plane crossing in a 2D histogram. To that end, we created a program that looked at a single slice of the accretion disk (in this case, a plane located at $\varphi = 0$, passing through the central body and the center of the donor star). In addition to allowing us to visualize all of the geodesic crossings simultaneously, we can use this histogram to get a clear idea

[9] One could, in theory, create a scattering orbit whose point of closest approach was within the event horizon tolerance. This particle would be reported as captured by the simulation, even though in reality it escaped to infinity. In practice, however, such orbits would be extremely hard to come by from variations of a circular orbit, especially in the presence of such a strong damping term.

[10] Due to an unfortunate error in the initial calculations, the Lagrangian for the circular orbit (equation 3.6) was set to -1, instead of the needed $-1/2$. This caused a miscalculation in the initial energy and standard deviation for the circular orbit of the donor star. Despite this, the mean orbit was still a circular orbit at the correct radius, and the perturbations were still physical (corresponding to the correct maxima and minima). Additionally, since the simulation was re-parameterized in coordinate time, no incorrect proper time effects were produced. In an unrelated error, the variation of the Carter constant was initially calculated to be 3 orders of magnitude too low. While this is undesirable, even at 5σ, this represents a difference in our initial $\dot{\theta}$ coordinate on the order of $\sim 10^{-12}$.

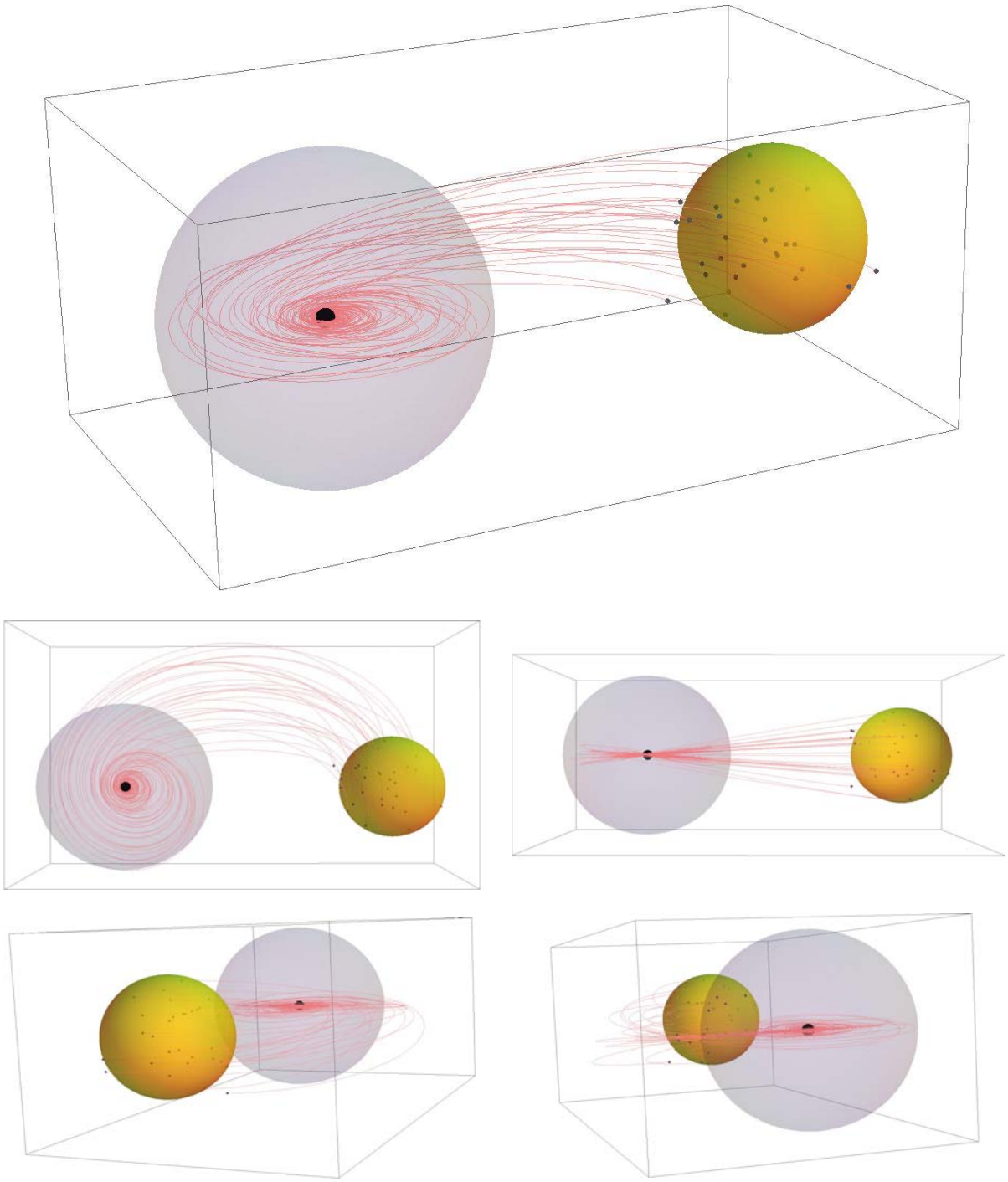

Figure 3.1: Example accretion disk orbits for GRS 1915+105. The particle geodesics (in red) are given initial conditions about the companion star (in yellow). The particles then travel about their perturbed circular orbit until they enter the damped region (in gray), at which point they lose energy and fall into the black hole (in black).

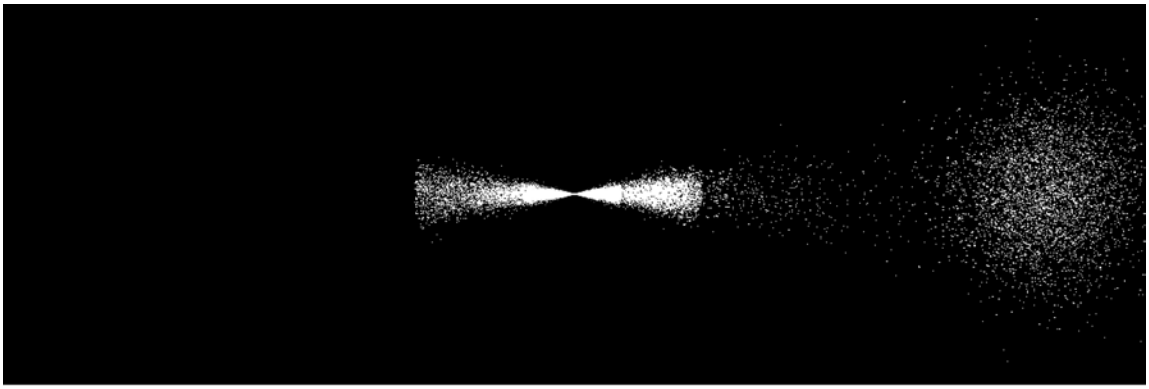

Figure 3.2: 1.2×10^8 $M_\odot$ by 4×10^7 $M_\odot$ cross section of the accretion disk geodesics, with each point representing a geodesic crossing the $\varphi = 0$ plane. This histogram covers nearly the full range of the simulation.

of the difference between the parallel and anti-parallel case.

Figure 3.2 shows the full range of the simulation for the parallel case. On the right, the donor star can be seen as a circular mass of points, representing the starting positions for the simulation. In the center, the disk thickness decreases as the particles are pulled towards the central body. From this perspective, it is easy to see that figure 3.2 is simply an angular slice of figure 3.1. With this technique, we can now zoom in on the strong field regime for the parallel and anti-parallel cases.

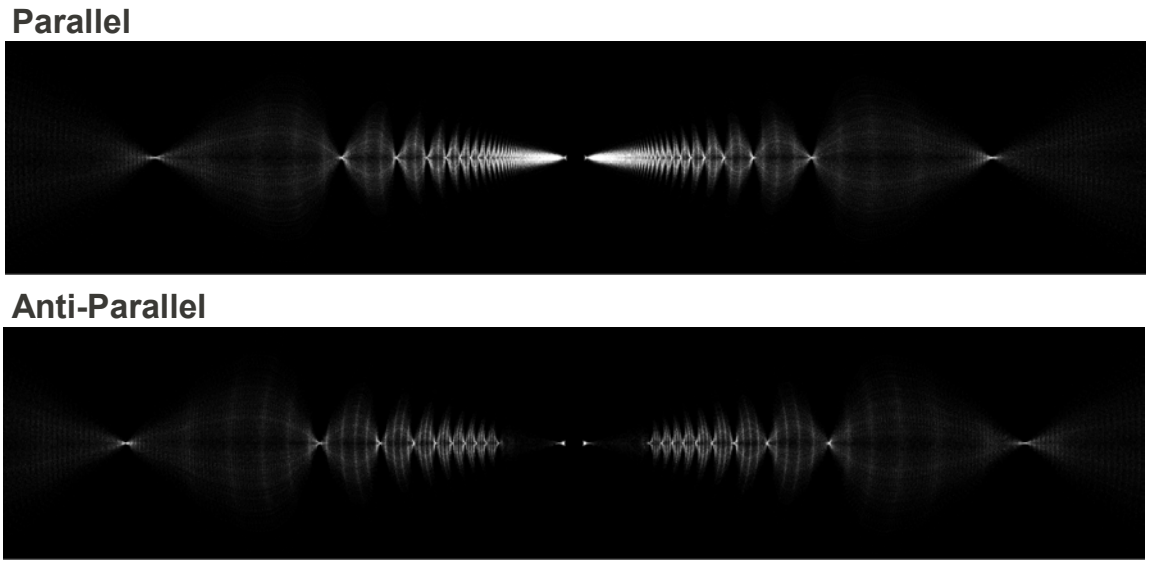

Figure 3.3: 4×10^3 $M_\odot$ by 600 $M_\odot$ cross section of the accretion disk geodesics for the parallel and anti-parallel cases, with each point representing a geodesic crossing the $\varphi = 0$ plane.

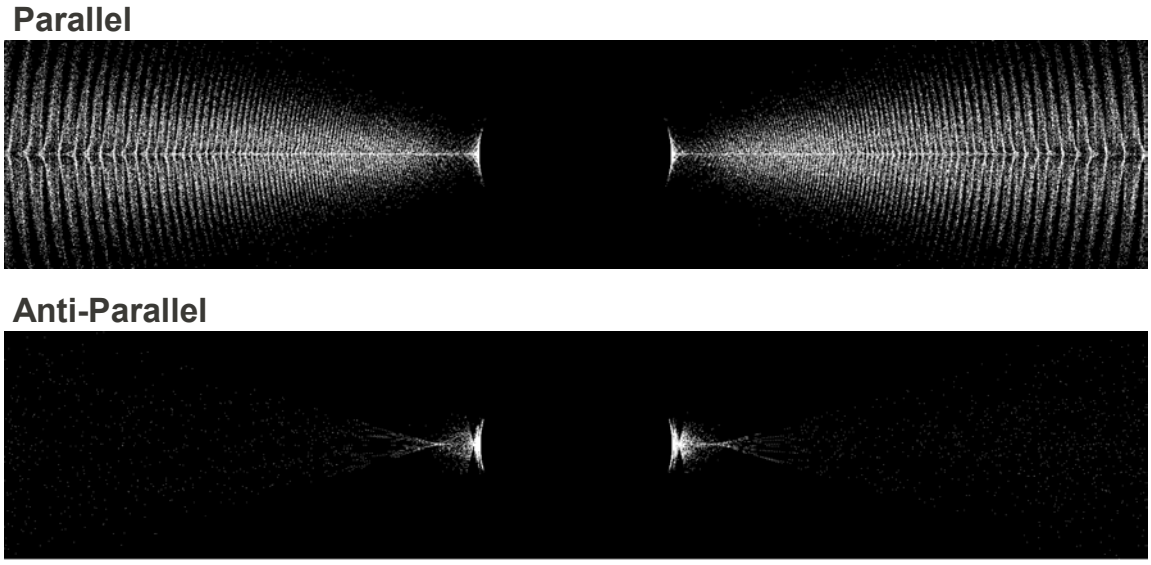

Figure 3.4: 200 $M_\odot$ by 40 $M_\odot$ cross section of the accretion disk geodesics for the parallel and anti-parallel cases, with each point representing a geodesic crossing the $\varphi = 0$ plane.

Looking to figure 3.3, we see that there is a clear difference between the two cases. As we approach the event horizon, the density of particles increases rapidly, exactly as we would expect. More and more particles get pulled into a decreasing circular orbit, slowly decreasing in distance from the equatorial plane as they approach the singularity. Contrast this with the behavior of the anti-parallel simulation: at first the density of particles increases inversely to radial distance, exactly as in the parallel case. But after a while the density of points drops off dramatically due to the particles loosing the majority of their angular velocity and falling into the black hole (see figure 3.4). Specifically, the region of decreased density extends radially from the center to a radius of approximately 126 $M_\odot$. This value corresponds *exactly* to the **innermost stable circular orbit** (ISCO). Defined as the minimum distance a particle of any energy can achieve a stable circular orbit, the anti-parallel ISCO for extreme Kerr is equal to nine times the mass of the central body.[11]

The fact that the anti-parallel case experiences a rapid decline in particle density at the ISCO suggests that the orbits are becoming circularized prior to infall. This is not observed for the parallel case where the ISCO is located nearly at the event horizon. This result represents a noticeable physical difference between mass transfer for systems with parallel and anti-parallel spins, something that could potentially be an observable phenomena.

What this result does *not* explain is the obvious periodicity in both cases. Initially it was suspected that this was simply a byproduct of the Gaussian distribution: if we

[11] Chandrasekhar, 1992, p. 341.

were simulating a stream of gas as a massive bundle of particles, then that stream should wrap around the star as a single tube of gas (as initially appears to be the case in figure 3.3). However, if the gas flow were to remain coherent (and there is no reason it should), we would expect any cross section of the gas to have a 2D Gaussian profile in the density. This is clearly not the case. In fact, the bumps in figures 3.3 and 3.4 are substantially less dense in the center, and seem to have much higher densities as the nodes between the periodicities.

A second possibility is that the nodes correspond to stable circular orbits in Kerr. Since stable orbits tend to be local attractors, pulling slightly perturbed particles towards them, it is conceivable that the nodes represent the local minima of the Kerr effective potential. Unfortunately, since there is no clear energy or angular momentum for the particles in bulk, it is unclear how these minima are generated, or how they are being effected by the damping term. It is possible that the particles are briefly pulled into a circular orbit, and are held there by the local minima, until the energy loss from the damping term forces them out again, on to the next circular semi-stable orbit. The fact that the ISCO is unaffected by the damping term may suggest that the circular orbits for Kerr are unaffected as well. Unfortunately, time did not permit us to test this.

Conclusion

For the most part, we have accomplished what we set out to do. Despite some slight errors in the calculation of the initial conditions, the simulations were completed as hoped. We were able to extract qualitative information about the nature of accretion disks in the Kerr geometry, and demonstrate strong field effects using our toy system (based on an observed astrophysical system, GRS 1915+105) that are unique to general relativity.

We first wrote a simulation to numerically compute the free fall geodesics for the Kerr geometry. Using the relativistic Lagrangian, we were able to write down second order equations of motion for Schwarzschild and Kerr, and then determine constants of the motion that were used to parameterize the system. These were embedded in a 54 Runge-Kutta algorithm which used the Kerr line element to maintain a local error tolerance at every step in the simulation. To test the program, we determined the orbital parameters for the planet Mercury and ran the simulation for a century of coordinate time. By recording the φ component of the orbit at each closest approach, we were able to numerically compute the precession of the perihelion to be 42.94 arcseconds per century, within 0.09% of the accepted value.

We then moved on to an exploration of the geodesics that could account for mass transfer and accretion disk structure in the Kerr geometry. By introducing a damping term into the geodesic equation at the classically predicted boundary for the accretion disk, we were able to force perturbed circular orbits from the companion star into decreasing circular orbits about the central body. Then by looking at the bulk behavior of this system, we were able to gain an understanding of what paths gas flow could take within the strong field regime. As expected, the frame dragging effects from the near extreme black hole caused a significant difference between the parallel and anti-parallel cases. This resulted from the theoretical difference for the innermost stable circular orbits between the two cases, and was represented quite clearly in our data.

The periodic nature of the accretion disk geodesics still remains to be explained. Our best theory is that the periodic nodes represent circular orbits in the Kerr geometry, and that the geodesics congregate in these minima of the effective potential until the energy loss from the damping term is sufficient to force them out again. What is unclear is what energy and angular momentum values are responsible for the placement of these semi-stable circular orbits. It should be interesting to see what values these orbits correspond to, and if they are influenced by the damping term.

Appendix A

Source Code

Here we reproduce the source code of our RK54 Algorithm used in this thesis. The code is written in C, compiled using the GNU C compiler. The compiler must link to the standard math libraries, as well as the GNU scientific libraries (for the random number distributions). Additionally, if compiling with optimizations, you must disable heuristic loop guessing, as the optimization disrupts the adaptive stepping protocol. Use the compiler flag "-fno-guess-branch-probability".

A.1 Header Program

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>

#define csc(x)          (1/sin((double)(x)))
#define cot(x)          (1/tan((double)(x)))
#define E               2.71828182845904523536029
#define PI              3.14159265358979323846264
#define PIover2         1.57079632679489661923132
#define SQRT2           1.41421356237309504880168
#define SOL             299792458.
#define SOLARMASS       1476.71618
#define GRAV_CONST      7.53462359e-21

/*The sigma values for 2 sigma*/
#define ENERGY_SIGMA    1.41421337/*0.207107*/
#define JZ_SIGMA        1690.01
#define CARTER_SIGMA    13313.1
#define THETA_SIGMA     0.0957049
#define R_SIGMA         4.9445e6


/*The Cash-Karp Parameters for Embedded Runga-Kutta*/
```

```
#define A_2 0.2
#define A_3 0.3
#define A_4 0.6
#define A_5 1.
#define A_6 0.875

#define B_21 0.2
#define B_31 0.075
#define B_32 0.225
#define B_41 0.3
#define B_42 (-0.9)
#define B_43 1.2
#define B_51 (-0.203703704)
#define B_52 2.5
#define B_53 2.592592593
#define B_54 (-1.296296296)
#define B_61 0.029495804
#define B_62 0.341796875
#define B_63 0.041594329
#define B_64 0.400345414
#define B_65 0.061767578

#define C_1 0.097883598
#define C_2 0
#define C_3 0.40257649
#define C_4 0.21043771
#define C_5 0
#define C_6 0.289102202

#define C_1STAR  0.102177373
#define C_2STAR 0
#define C_3STAR 0.383907903
#define C_4STAR 0.244592737
#define C_5STAR 0.019321987
#define C_6STAR 0.25

#define CDIFF_1 (-0.004293775)
#define CDIFF_2 0
#define CDIFF_3 0.018668587
#define CDIFF_4 (-0.034155027)
#define CDIFF_5 (-0.019321987)
#define CDIFF_6 0.039102202
```

A.2 Core Program

```
#include "RKheader.h"

/*Converts from the relativistic constants of motion (energy, spin, and carter's constant)
 to the starting coordinates for the simulation*/
int convert
```

```
(
  double energy,
  double jz,
  double carter,
  double M,
  double a,
  double *f
){
 double test;

  f[1] = -energy - (2*M*f[2]*(a*(a*energy + jz) + energy*pow(f[2],2)))/((pow(a,2)
         *pow(cos(f[4]),2) + pow(f[2],2))*(pow(a,2) - 2*M*f[2] + pow(f[2],2)));
  f[5] = sqrt(2*a*energy*jz + carter - pow(jz,2)*pow((1/sin(f[4])),2) + pow(a,2)
         *(pow(cos(f[4]),2) - pow(energy,2)*pow(sin(f[4]),2)))/ pow(pow(f[2],2) +
         pow(a,2)*pow(cos(f[4]),2),2);
  test = 2*a*energy*jz + carter - pow(jz,2)*pow((1/sin(f[4])),2) + pow(a,2)*(pow(
         cos(f[4]),2) - pow(energy,2)*pow(sin(f[4]),2));
  if(test <= 0.)f[5] = 0.;
    /*shody piece of chicanery; it handles floating point errors that somtimes
      conspire to give an imaginary theta_dot for planar orbits*/
  f[7] = (pow(a,2)*jz*pow((1/tan(f[4])),2) + f[2]*(-2*a*energy*M + jz*pow((1/sin(
         f[4])),2)*(-2*M + f[2])))/((pow(a,2)*pow(cos(f[4]),2) + pow(f[2],2))*(
         pow(a,2) - 2*M*f[2] + pow(f[2],2)));
  return 0;
}

/*The first order vector form of the Kerr Lagrangian (divided by tdot to parameterize in
coordinate time)*/
int func
(
  double t,
  double tdot,
  double r,
  double rdot,
  double th,
  double thdot,
  double ph,
  double phdot,
  double *g,
  double M,
  double a,
  double b,
  double disk_bound
){
  g[0] = 1;
  g[1] = (M*rdot*(2.*(pow(a,2.) + pow(r,2.))*tdot*(-pow(r,2.) + pow(a,2.)*pow(
         cos(th),2.)) + a*phdot*(6.*pow(r,4.) - 2.*pow(a,4.)*pow(cos(th),2.) +
         pow(a,2.)*pow(r,2.)*(3. + cos(2.*th)))*pow(sin(th),2.)) - 2.*pow(a,2.)
         *M*r*(pow(a,2.) - 2.*M*r + pow(r,2.))*thdot*(-tdot + a*phdot*pow(sin(
         th),2.))*sin(2.*th))/((pow(a,2.) - 2.*M*r + pow(r,2.))*tdot*pow(pow(r,
         2.) + pow(a,2.)*pow(cos(th),2.),2.));
  g[2] = rdot/tdot;
  g[3] = -((pow(a,2.) - 2.*M*r + pow(r,2.))*((-2.*r*pow(rdot,2.))/(pow(a,2.) -
         2.*M*r + pow(r,2.)) - 2.*r*pow(thdot,2.) + (2.*M*pow(tdot,2.)*(pow(r,2.)
```

```
        - pow(a,2.)*pow(cos(th),2.))) / pow(pow(r,2.) + pow(a,2.)*pow(cos(th),
        2.),2.) + (2.*(M - r)*pow(rdot,2.)*(pow(r,2.) + pow(a,2.)*pow(cos(th),2.
        ))) / pow(pow(a,2.) - 2.*M*r + pow(r,2.),2.) - (8.*a*M*phdot*pow(r,2.)*
        tdot*pow(sin(th),2.)) / pow(pow(r,2.) + pow(a,2.)*pow(cos(th),2.),2.) +
        (4.*a*M*phdot*tdot*pow(sin(th),2.))/(pow(r,2.) + pow(a,2.)*pow(cos(th),2.
        )) - (pow(phdot,2.)*pow(sin(th),2.)*(4.*pow(r,3.) + pow(a,2.)*r*(3. +
        cos(2.*th)) + 2.*pow(a,2.)*M*pow(sin(th),2.)))/(pow(r,2.) + pow(a,2.)*
        pow(cos(th),2.)) + (2.*pow(phdot,2.)*r*pow(sin(th),2.)*(pow(pow(a,2.) +
        pow(r,2.),2.) - pow(a,2.)*(pow(a,2.) - 2.*M*r + pow(r,2.))*pow(sin(th),2.
        ))) / pow(pow(r,2.) + pow(a,2.)*pow(cos(th),2.),2.) -(2*rdot*(-2*r*rdot +
        pow(a,2)*thdot*sin(2*th)))/(pow(a,2) - 2*M*r + pow(r,2)))/(2.*tdot*(pow(
        r,2.)+ pow(a,2.)*pow(cos(th),2.))) - (1./(1.+exp(1.*(r - disk_bound))))
        *b*rdot/r;
    g[4] = thdot/tdot;
    g[5] = -((2.*r*rdot*thdot + ((-(pow(phdot,2.)*pow(pow(a,2.) - 2.*M*r + pow(r,
           2.),2.)) + pow(a,2.)*(pow(rdot,2.) - (pow(a,2.) - 2.*M*r + pow(r,2.))*
           pow(thdot,2.)))*sin(2.*th)) / (2.*(pow(a,2.) - 2.*M*r + pow(r,2.))) -
           (M*r*pow(phdot*(pow(a,2.) + pow(r,2.)) - a*tdot,2.)*sin(2.*th))/pow(pow
           (r,2.) + pow(a,2.)*pow(cos(th),2.),2.))/(tdot*(pow(r,2.) + pow(a,2.)*
           pow(cos(th),2.)))) - (1./(1.+exp(1.*(r - disk_bound))))*b*thdot/r;
    g[6] = phdot/tdot;
    g[7] = -(csc(th)*(-4.*a*M*tdot*(8.*r*(pow(a,2.) - 2.*M*r + pow(r,2.))*thdot*
           cos(th) + 4.*rdot*(-pow(r,2.) + pow(a,2.)*pow(cos(th),2.))*sin(th)) +
           phdot*(16.*(pow(a,2) - 2.*M*r + pow(r,2.))*thdot*cos(th)*(pow(pow(r,2.)
           + pow(a,2.)*pow(cos(th),2.),2.) + 2.*pow(a,2.)*M*r*pow(sin(th),2.)) +
           4.*rdot*sin(th)*(-8.*M*pow(r,4.) + 4.*pow(r,5.) + 8.*pow(a,2.)*pow(r,3.)
           *pow(cos(th),2.) + 4.*pow(a,4.)*r*pow(cos(th),4.) - 2.*pow(a,2.)*M*pow
           (r,2.)*(3. + cos(2.*th)) + pow(a,4.)*M*pow(sin(2.*th),2.))))) / (8.*(
           pow(a,2.) - 2.*M*r + pow(r,2.))*tdot*pow(pow(r,2.) + pow(a,2.)*pow(
           cos(th),2.),2.)) - (1./(1.+exp(1.*(r - disk_bound))))*b*phdot/r;

return 0;
}

double kerr_error
(
  double *f_temp,
  double *error,
  double M,
  double a
){
  double proper_error;
  double proper_v_error;

  proper_error = (-1 + (2*M*f_temp[2])/(pow(f_temp[2],2) + pow(a,2)*pow(cos(
         f_temp[4]),2)))*pow(error[0],2) + ((pow(f_temp[2],2) + pow(a,2)*pow(cos(
         f_temp[4]),2))/(pow(a,2) - 2*M*f_temp[2] + pow(f_temp[2],2)))*pow(error[2],
         2) + (pow(f_temp[2],2) + pow(a,2)*pow(cos(f_temp[4]),2))*pow(error[4], 2) +
         ((pow(sin(f_temp[4]),2)*(pow(pow(a,2) + pow(f_temp[2],2),2) - pow(a,2)*(
         pow(a,2) - 2*M*f_temp[2] + pow(f_temp[2],2))*pow(sin(f_temp[4]),2)))/(pow(
         f_temp[2],2) + pow(a,2)*pow(cos(f_temp[4]),2)))*pow(error[6], 2) + ((-2*a*M
         *f_temp[2]*pow(sin(f_temp[4]),2))/(pow(f_temp[2],2) +  pow(a,2)*pow(cos(
         f_temp[4]),2)))*error[0]*error[6];
```

```
  proper_v_error = (-1 + (2*M*f_temp[2])/(pow(f_temp[2],2) + pow(a,2)*pow(cos(
         f_temp[4]),2)))*pow(error[1],2) + ((pow(f_temp[2],2) + pow(a,2)*pow(cos(
         f_temp[4]),2))/(pow(a,2) - 2*M*f_temp[2] + pow(f_temp[2],2)))*pow(error[3],
         2) + (pow(f_temp[2],2) + pow(a,2)*pow(cos(f_temp[4]),2))*pow(error[5], 2) +
         ((pow(sin(f_temp[4]),2)*(pow(pow(a,2) + pow(f_temp[2],2),2) - pow(a,2)*(
         pow(a,2) - 2*M*f_temp[2] + pow(f_temp[2],2))*pow(sin(f_temp[4]),2)))/(pow(
         f_temp[2],2) + pow(a,2)*pow(cos(f_temp[4]),2)))*pow(error[7], 2) + ((-2*a*M
         *f_temp[2]*pow(sin(f_temp[4]),2))/(pow(f_temp[2],2) + pow(a,2)*pow(cos(
         f_temp[4]),2)))*error[1]*error[7];

  return sqrt(pow(proper_error, 2.) + pow(proper_v_error, 2.));
}

int main(int argc, char **argv)
{
  double M;
  double a;
  double b;
  double disk_bound;
  double energy;
  double energy_temp;
  double jz;
  double phdot_central;
  double carter;
  double carter_min;
  double *f;
  double *g;
  double *f_temp;
  double *error;
  double ds;
  double ds_old;
  double t;
  double dt;
  double input_error;
  double proper_error;
  double proper_v_error;
  double total_error;
  double event_horizon;
  double horizon_tolerance;
  double t_max;
  double k1[8], k2[8], k3[8], k4[8], k5[8], k6[8];
  int i, j;
  long NN;
  FILE *filefull, *filepolar;
  char Kerrfull[20], Kerrpolar[20];

  /*Sets up the GSL random number generator*/
  const gsl_rng_type * T;
  gsl_rng * r;
  T = gsl_rng_default;
  gsl_rng_env_setup();
  r = gsl_rng_alloc (T);

  /*handles the in/out vectors as pointers for purposes
```

```
 of function passing*/
g = (double *)malloc(8.*sizeof(double));
f = (double *)malloc(8.*sizeof(double));
f_temp = (double *)malloc(8.*sizeof(double));
error = (double *)malloc(8.*sizeof(double));
for( i = 0 ; i < 8 ; i++){
  g[i] = 0.;
  f[i] = 0.;
  f_temp[i] = 0.;
  error[i] = 0.;
  k1[i] = 0.;
  k2[i] = 0.;
  k3[i] = 0.;
  k4[i] = 0.;
  k5[i] = 0.;
  k6[i] = 0.;
}

/*defauly error and event horizon tolerance*/
input_error = 0.00001;
horizon_tolerance = 0.01;
dt = 1.;
t = 0.;
i = 0;
j = 0;
f[4] = PIover2;
NN = 0;
b = 0.;

/*The input/output for the command line*/
while(i < argc){
  if ( strcmp(argv[i],  "--M") == 0 )
    M = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--a") == 0 )
    a = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--b") == 0 )
    b = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--db") == 0 )
    disk_bound = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--r") == 0 )
    f[2] = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--e") == 0 )
    energy = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--jz") == 0 )
    jz = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--k") == 0 )
    carter = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--th") == 0 )
    f[4] = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--ph") == 0 )
    f[6] = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--ie") == 0 )
    input_error = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--eh") == 0 )
```

```
    horizon_tolerance = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--tm") == 0 )
    t_max = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--dt") == 0 )
    dt = atof(argv[++i]);
  else if ( strcmp(argv[i],  "--nn") == 0 )
    NN = atoi(argv[++i]);
  else if ( strcmp(argv[i],  "--help") == 0 ){
    fprintf(stderr, "\nKerr Geodesic Simulator V2.0\n"
      "Author: Carl Rodriguez\n\n"
      "Uses the relativistic lagrangian and the full kerr metric \n"
      "to compute the trajectories of a test body with an RK54 ODE solver\n\n"
      "Usage: --M  mass of central body\n"
      "   --a   spin of central body\n"
      "   --b   damping coefficient (default is 0)\n"
      "   --db  disk boundary\n"
      "   --r   initial radial position\n"
      "   --e   orbital energy\n"
      "   --jz  orbital spin\n"
      "   --k   Carter's constant\n"
      "   --th  inital theta position (default is pi/2)\n"
      "   --ie  Error epsilon (default is 1e-5)\n"
      "   --ph  initial phi position (default is 0)\n"
      "   --eh  Event Horizon epsilon (default is 0.01)\n"
      "   --dt  Time Intervals (default is 1m)\n"
      "   --tm  Max run time (in coordinate time)\n"
      "   --nn  Run/RNG seed number (for cluster)\n");
      exit(0);}
    i++;
  }

sprintf(Kerrfull, "Kerrfull_%ld.dat", NN);
sprintf(Kerrpolar, "Kerrpolar_%ld.dat", NN);

gsl_rng_set (r, NN);

/*Adds gaussian variation to the input parameters, then converts them to
 spherical initial conditions. Also checks to make sure Carter's Constant
 is physical*/

if(NN != 0){
  phdot_central = (4*(pow(f[2],2) + pow(a,2)*pow(cos(f[4]),2))*pow(csc(f[4]),2)*
    (jz*f[2]*(-2*M + f[2]) + a*(a*jz*pow(cos(f[4]),2) + 2*energy*M*f[2]*pow(sin(
    f[4]),2)))) / ((pow(a,2) + f[2]*(-2*M + f[2]))*pow(pow(a,2) + 2*pow(f[2],2)
    + pow(a,2)*cos(2*f[4]),2));
  do energy_temp = gsl_ran_gaussian(r, ENERGY_SIGMA);
    while(energy_temp + energy <= 0.);
  energy += energy_temp;
  f[4] += gsl_ran_gaussian(r, THETA_SIGMA);
  f[2] += gsl_ran_gaussian(r, R_SIGMA);
  jz = ((-4*a*f[2]*energy*M + (pow(a,2) + 2*pow(f[2],2))*(pow(a,2) + f[2]*(f[2] -
       2*M))*phdot_central + pow(a,2)*(pow(a,2) + f[2]*(f[2] -2*M))*phdot_central
       *cos(2*f[4]))*pow(sin(f[4]),2))/(pow(a,2) + 2*f[2]*(f[2] - 2*M) + pow(a,2)
       *cos(2*f[4]));
```

```
  carter += gsl_ran_gaussian(r, CARTER_SIGMA);
}

fprintf(stderr, "RNG Seed:%ld R:%lf Jz:%lf e:%lf theta:%lf K:%lf\n",NN,
f[2],jz,energy,f[4],carter);
carter_min = -2.*a*energy*jz - pow(a,2.)*pow(cos(f[4]),2.) + pow(jz,2.)*pow(csc(
       f[4]),2.) + pow(a,2.)*pow(energy,2.)*pow(sin(f[4]),2.);
if(carter < carter_min){
    fprintf(stderr,"WARNGING: Input Carter's Constant is below possible minimum.\n"
                  "Setting constant to %0.15lf\n", carter_min);
    carter = carter_min;
}

convert(energy, jz, carter, M, a, f);

filefull = fopen(Kerrfull, "w");
filepolar = fopen(Kerrpolar, "w");

ds = input_error;
event_horizon = M + sqrt(pow(M,2.) - pow(a,2.));
i = 0;

/*RK54 ODE Solver*/
while(f[0] <= t_max){
  while(f[0] >= t + dt ){
    t += dt;
    energy = -(pow(a,2)*pow(cos(f[4]),2)*f[1] + f[2]*(f[1]*(-2*M + f[2]) + 2*a*M
             *f[7]*pow(sin(f[4]),2)))/(pow(a,2)*pow(cos(f[4]),2) + pow(f[2],2));
    jz = (pow(sin(f[4]),2)*(-2*a*M*f[1]*f[2] + pow(pow(a,2) + pow(f[2],2),2)*f[7]
         - pow(a,2)*(pow(a,2) + f[2]*(-2*M + f[2]))*f[7]*pow(sin(f[4]),2)))/(pow(
         a,2)*pow(cos(f[4]),2) + pow(f[2],2));
    fprintf(filefull, "%lf     %lf     %lf     %lf %0.12lf %lf\n",f[0], f[2],
         f[4], f[6], energy, jz);
  }

  fprintf(filepolar, "%lf %lf %lf\n", f[2], f[4], f[6]);
  if( i == 0 && f[2] < disk_bound){
    fprintf(stderr, "Disk Geodesic!\n");
    fprintf(stdout, "%ld\n", NN);
    ++i;
  }

  do{
    func(f[0],f[1],f[2],f[3],f[4],f[5],f[6],f[7],g,M,a,b,disk_bound);
    for(j = 0 ; j < 8 ; j++) k1[j] = ds * g[j];
    func(f[0]+B_21*k1[0],f[1]+B_21*k1[1],f[2]+B_21*k1[2],f[3]+B_21*k1[3],f[4]+B_21
        *k1[4],f[5]+B_21*k1[5],f[6]+B_21*k1[6],f[7]+B_21*k1[7],g,M,a,b,disk_bound);
    for(j = 0 ; j < 8 ; j++) k2[j] = ds * g[j];
    func(f[0]+B_31*k1[0]+B_32*k2[0],f[1]+B_31*k1[1]+B_32*k2[1],f[2]+B_31*k1[2]+
        B_32*k2[2],f[3]+B_31*k1[3]+B_32*k2[3],f[4]+B_31*k1[4]+B_32*k2[4],f[5]+
        B_31*k1[5]+B_32*k2[5],f[6]+B_31*k1[6]+B_32*k2[6],f[7]+B_31*k1[7]+B_32*k2[7]
        ,g,M,a,b,disk_bound);
    for(j = 0 ; j < 8 ; j++) k3[j] = ds * g[j];
    func(f[0]+B_41*k1[0]+B_42*k2[0]+B_43*k3[0],f[1]+B_41*k1[1]+B_42*k2[1]+B_43*k3[1],
```

```
          f[2]+B_41*k1[2]+B_42*k2[2]+B_43*k3[2],f[3]+B_41*k1[3]+B_42*k2[3]+B_43*k3[3],
          f[4]+B_41*k1[4]+B_42*k2[4]+B_43*k3[4],f[5]+B_41*k1[5]+B_42*k2[5]+B_43*k3[5],
          f[6]+B_41*k1[6]+B_42*k2[6]+B_43*k3[6],f[7]+B_41*k1[7]+B_42*k2[7]+B_43*k3[7],
          g,M,a,b,disk_bound);
      for(j = 0 ; j < 8 ; j++) k4[j] = ds * g[j];
      func(f[0]+B_51*k1[0]+B_52*k2[0]+B_53*k3[0]+B_54*k4[0],f[1]+B_51*k1[1]+B_52*k2[1]
          +B_53*k3[1]+B_54*k4[1],f[2]+B_51*k1[2]+B_52*k2[2]+B_53*k3[2]+B_54*k4[2],f[3]
          +B_51*k1[3]+B_52*k2[3]+B_53*k3[3]+B_54*k4[3],f[4]+B_51*k1[4]+B_52*k2[4]+B_53
          *k3[4]+B_54*k4[4],f[5]+B_51*k1[5]+B_52*k2[5]+B_53*k3[5]+B_54*k4[5],f[6]+B_51*
          k1[6]+B_52*k2[6]+B_53*k3[6]+B_54*k4[6],f[7]+B_51*k1[7]+B_52*k2[7]+B_53*k3[7]
          +B_54*k4[7],g,M,a,b,disk_bound);
      for(j = 0 ; j < 8 ; j++) k5[j] = ds * g[j];
      func(f[0]+B_61*k1[0]+B_62*k2[0]+B_63*k3[0]+B_64*k4[0]+B_65*k5[0],f[1]+B_61*k1[1]+
          B_62*k2[1] + B_63*k3[1]+B_64*k4[1]+B_65*k5[1],f[2]+B_61*k1[2]+B_62*k2[2]+B_63*
          k3[2]+B_64*k4[2]+B_65*k5[2],f[3]+B_61*k1[3]+B_62*k2[3]+B_63*k3[3]+B_64*k4[3]+
          B_65*k5[3],f[4]+B_61*k1[4]+B_62*k2[4]+ B_63*k3[4]+B_64*k4[4]+B_65*k5[4],f[5]+
          B_61*k1[5]+B_62*k2[5]+B_63*k3[5]+B_64*k4[5]+B_65*k5[5],f[6]+B_61*k1[6]+B_62*
          k2[6]+B_63*k3[6]+B_64*k4[6]+B_65*k5[6],f[7]+B_61*k1[7]+B_62*k2[7]+B_63*k3[7]+
          B_64*k4[7]+B_65*k5[7],g,M,a,b,disk_bound);
      for(j = 0 ; j < 8 ; j++){
        k6[j] = ds * g[j];
        error[j] = CDIFF_1*k1[j] + CDIFF_3*k3[j] + CDIFF_4*k4[j] +
        CDIFF_5*k5[j] + CDIFF_6*k6[j];
        f_temp[j] = f[j] + k1[j]*C_1 + k3[j]*C_3 + k4[j]*C_4 + k6[j]*C_6;
      }

      total_error = kerr_error(f_temp, error, M, a);
      ds_old = ds;
      ds *= pow(input_error / total_error, 0.2);

    }while(ds_old > ds);

    for(j = 0 ; j < 8 ; j++) f[j] = f_temp[j];

    if(f[2] - event_horizon < horizon_tolerance ){
      fprintf(stderr, "INSPIRAL!\n");
      break;
    }
  }

  fclose(filefull);
  fclose(filepolar);

  if(i==0){
    remove(Kerrfull);
    remove(Kerrpolar);
  }

return 0;
}
}
```

A.3 Histogram Program

Reads in the coordinate time output files labeled "Kerrfull_().dat", and places a mark in each point that falls within dph of the selected azimuthal angle. The parenthesis are file numbers taken from a list of the disk geodesics labeled "interesting.full.dat".

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#define PI 3.14159265
#define TWOPI 6.283185307

int main(int argc, char **argv)
{
int a;
double **histogram;
double radial;
double height;
double phi_slice;
double phitolerance;
double phitolOVERtwo;
double halfheight;
long i;
long j;
long r_num;
long r_bin;
long z_num;
long z_bin;
double r_cyn;
double z;
double dr;
double dz;
double r;
double th;
double ph;
FILE *file;
FILE *data;
FILE *output;
char filename[18];

i = 0;
while(i < argc){
    if ( strcmp(argv[i],    "--r") == 0 )
         radial = atof(argv[++i]);
    else if ( strcmp(argv[i],       "--h") == 0 )
         height = atof(argv[++i]);
    else if ( strcmp(argv[i],    "--ph") == 0 )
         phi_slice = atof(argv[++i]);
    else if ( strcmp(argv[i],       "--dr") == 0 )
         dr = atof(argv[++i]);
    else if ( strcmp(argv[i],       "--dz") == 0 )
         dz = atof(argv[++i]);
```

```
    else if ( strcmp(argv[i],       "--dph") == 0 )
         phitolerance = atof(argv[++i]);
    else if ( strcmp(argv[i], "--help") == 0 ){
         fprintf(stderr, "\nStrong Field Cross Section Plotter\n"
                         "V1.0  Author: Carl Rodriguez\n\n"
                         "Calculates an azimuthal subsection of an accretion disk\n"
 "Designed for the near field regime, close to the central body\n\n"
                         "--r Radius of interest\n"
                         "--h Height of interest\n"
                         "--ph Location of phi slice (0 to pi)\n"
                         "--dr Radial bin size\n"
                         "--dz Height bin size\n"
                         "--dph Thickness of phi slice\n");
        exit(0);}
    i++;
}

halfheight = height/2.;
phitolOVERtwo = phitolerance/2.;

r_num = (long)(ceil((2*radial)/dr));
z_num = (long)(ceil(height/dz));

histogram = (double **)malloc(z_num * sizeof(double *));
for( i = 0 ; i < z_num ; i++)
    histogram[i] = (double *)malloc(r_num * sizeof(double));

for( i = 0 ; i < z_num ; i++ )
for( j = 0 ; j < r_num ; j++ )
   histogram[i][j] = 0;

file = fopen("interesting.full.dat","r");
if (file==0) {
    fprintf(stderr, "Cannot open interesting file list, aborting\n");
    return 0;
}

while (!feof(file)) {
    fscanf(file, "%d\n", &a);
    sprintf(filename, "Kerrpolar_%d.dat", a);
    data = fopen(filename, "r");
    if (data==0) {
        fprintf(stderr, "Cannot open %s, skipping\n", filename);
    }
    else{
        while (!feof(data)) {
            fscanf(data, "%lf       %lf     %lf\n", &r, &th, &ph);
            if (fmod(fabs(ph-phitolOVERtwo+phi_slice),TWOPI) < phitolerance){
                r_cyn = radial + (r * sin(th));
                z = r * cos(th) + halfheight;
                r_bin = (long)floor(r_cyn / dr);
                z_bin = (long)floor(z / dz);
                if((fabs(r_bin) < r_num) && (z_bin < z_num) && (r_bin >= 0 )
                  && (z_bin >= 0))
```

```
                histogram[z_bin][r_bin] += 1;
            }
            else if(fmod(fabs(ph-phitolOVERtwo+phi_slice) + PI,TWOPI)
              < phitolerance){
                r_cyn = radial - (r * sin(th));
                z = r * cos(th) + halfheight;
                r_bin = (long)floor(r_cyn / dr);
                z_bin = (long)floor(z / dz);
                if((fabs(r_bin) < r_num) && (z_bin < z_num) && (r_bin >= 0 )
                  && (z_bin >= 0))
                    histogram[z_bin][r_bin] += 1;
            }
        }
    }
fclose(data);
}
fclose(file);

output = fopen("StrongHistogram.dat", "w");

for(i = 0 ; i < z_num ; i++){
for(j = 0 ; j < r_num ; j++)
    fprintf(output, "%lf\t", histogram[i][j]);
    fprintf(output, "\n");
}

fclose(output);
return 0;
}
```

References

Paul Baker. *Electrodynamics and an Investigation of Weak Field Kerr Geometry.* Reed College Senior Thesis, 2006

Michael Bakich. *The Cambridge Planetary Handbook.* Cambridge University Press, 2000.

Bradley Carroll and Dale Ostlie. *An Introduction to Modern Astrophysics.* Pearson Education, Inc., 2007.

Subrahmanyan Chandrasekhar. *The Mathematica Theory of Black Holes.* Oxford University Press, 1992.

Joel Franklin. *Differential Equations & Numerical Solutions.* Scientific Computing Lecture 1, Reed College, 2009.

Jochen Greiner. "The Binary Components of GRS 1915+105." ArXiv:astro-ph/0111540v1, November 2001.

James Hartle. *Gravity: An Introduction to Einstein's General Relativity.* Pearson Education, Inc., 2003.

Jeffrey McClintock, Rebeca Shafee, Ramesh Narayan, Ronald Remillard, Shane Davis, and Li-Xin Li. "The Spin of the Near-Extreme Kerr Black Hole GRS 1915+105." ApJ, 652:518-539, November 2006.

William Press, Saul Teukolsky, William Vellerling, and Brian Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1988.

Edwin Taylor and John Wheeler. *Exploring Black Holes: Introduction to General Relativity.* Addison Wesley Longman, Inc., 2000.

Matt Visser. "The Kerr Spacetime: A Brief Introduction." ArXiv:gr-qc/0706.0622v3, January 2008.