Implementing covariant Monte Carlo radiation transfer in Athena++

Robin Leichtnam
Advisor: Shane Davis
May 2019

Department of Astronomy
University of Virginia

This thesis is submitted in partial completion of the requirements of the
Bachelor of Science Astronomy-Physics Major

Draft version May 8, 2019
Preprint typeset using LaTeX style emulateapj v. 12/16/11

IMPLEMENTING COVARIANT MONTE CARLO RADIATION TRANSFER IN ATHENA++

Robin Leichtnam[1]
[1]Dept. of Astronomy, University of Virginia, Charlottesville, Virginia 22904, USA
Draft version May 8, 2019

ABSTRACT

With the development of computational astrophysics, theoretical study and modeling of astronomical objects has become a leading field in astronomy. An important advancement came with the ability to study black holes and other compact objects by creating mock images and spectra that can be compared to observations, facilitating the study of these bodies, most of which cannot be directly imaged with the current technology. Athena++ is an astrophysical magnetohydrodynamics (MHD) code written in C++, which coupled with a Monte Carlo radiation transfer, can be used to create and postprocess snapshots of numerical simulations and generate spectra. For the simulation of accretion disks around black holes, we need to account for the strong gravity generated by the massive central object. We implemented a geodesic integration algorithm in the Athena++ code to solve the covariant radiation algorithm. The method works for a general choice of coordinates and spacetime metric and tests are presented for several choices of coordinates and spacetimes: Kerr-Schild, spherical-polar, and cylindrical.

1. INTRODUCTION

Black holes are some of the most fascinating objects in the universe. Resulting from the death of stars, they are created by the gravitational collapse of the remains of the dead stars. The pressure being unable to counteract the gravitational force, matter falls on itself and creates a point like singularity with a mass that can be up to millions of time the mass of the Sun. This results in a gravitational force powerful enough to prevent light from escaping a region inside what is called the Schwarzschild radius, which is given by:

$$R = \frac{2GM}{c^2} \tag{1}$$

This corresponds to a radius of approximately 3 km for a black hole of 1 $M_\odot$, making it impossible to resolve any of the known black holes with today's telescope, with the only exception of M87 which was imaged recently with the Event Horizon Telescope. However, we can still obtain information about black holes by analyzing the spectra emitted by accretion disks around them. Those disks are thought to be geometrically thin but optically thick and are often assumed to produce blackbody emission.

While spectra can be taken with observations, it is important to be able to compare the data to theoretical models. We apply the Monte Carlo method to the radiation in accretion disks, to determine for example the initial position and direction of photons as well as their scattering and absorption. This method uses random sampling to solve deterministic problems and obtain numerical results. Simulations based on radiative transfer and using a Monte Carlo method then provide the means to create models with adjustable physical parameters. Changing these initial conditions produce different spectra which we can then compare to observations, giving information about the observed accretion disks and their corresponding black holes when the models and the observations match.

Simulations of accretion disks are therefore necessary to understand black holes due to the difficulty of observing them directly. Spectra are created using radiative transfer which calculates the path of photons in a grid while taking into account different parameters such as opacity, scattering, and absorption. This grid can be made to fit various coordinate systems such as cartesian, cylindrical, and spherical for a uniform sphere. We recently added to the code an algorithm in spherical coordinates for a non-uniform sphere. However, black holes are the most exotic astronomical objects and create such a strong gravity that a classical approximation for the radiative transfer cannot always be used. These sets of coordinates only work in flat spacetime, and while this is a good approximation far from the black hole, it becomes unrealistic as the photons get closer to it where spacetime is strongly curved.

Because of these extreme conditions, the use of general relativity becomes essential. The path of photons around a black hole cannot simply be calculated using classical mechanics but needs to be computed as geodesics. Being the equivalent of straight line in curved spacetime, geodesics represent the path of a particle. They are particularly important in general relativity because they are computed using the four dimensions necessary to the definition of curved spacetime: time along with the three usual spatial dimensions. Those paths are calculated using the following equation:

$$\frac{d^2x^\lambda}{dt^2} + \Gamma^\lambda_{\mu\nu}\frac{dx^\mu}{dt}\frac{dx^\nu}{dt} = 0 \tag{2}$$

where Γ is the Christoffel symbols of the metric and λ, μ, and ν are the coordinate numbers. This expression follows the Einstein summation convention.

Geodesics provide a compact way to describe the trajectory of a particle in a four dimensional space under the influence of gravity. Moreover, these equations can replace the previous algorithms used because geodesics, when computed with the corresponding metrics, give the

right result for flat spacetime too. This also provides a way to check the veracity of the implemented algorithm by comparing the results to previous ones obtained from classical coordinate systems.

We first present the Athena++ code and radiation transfer in Section 2. We then give the method used for the implementation in Section 3 and describe the implementation for Kerr-Schild, spherical-polar, and cylindrical coordinates along with their accuracy checks in Section 4. Finally, we conclude in Section 5.

2. RADIATION TRANSFER IN ATHENA++

2.1. *Athena++*

Athena++ is an astrophysical magnetohydrodynamics (MHD) code written in C++. It is a state-of-the-art code used for a multitude of applications and simulations, from accretion disks around black holes to star and planet formation. While it is very advancement regarding technicalities such as computational speed and parallelization, it becomes incredibly useful for simulations thanks to its grid options including adaptive mesh refinement (AMR). AMR allows the user to assign a better precision to specific areas in the grid while keeping the computation optimal by also allowing the grid to be non-uniform and therefore less precise in areas of lesser interest. With this, we can focus on specific parts of the simulation and use large dynamic range while keeping the computational expense to a minimum. Athena++ also includes general relativistic MHD (GRMHD), further expanding the range of application of the code.

However, the Athena++ treatment of radiation transfer is frequency averaged meaning that no spectral information can be obtained directly from simulations ran using this code. In order to create mock spectra that we could compare with observations, we need to implement a radiation transfer code which is frequency dependent.

2.2. *Monte Carlo method for radiation transfer*

This radiation transfer code uses the Monte Carlo method, a method based on the use of random sampling to compute numerical values. It propagates photons using probabilities and calculates the distance that a photon will do before being either absorbed or scattered using a probability distribution function as follow:

$$\psi(x_0) = \frac{\int_a^{x_0} P(x)dx}{\int_a^b P(x)dx} \tag{3}$$

where $\psi(x_0)$ ranges from 0 to 1 as x_0 goes from a to b. We can first find a "uniform random deviate" called ξ, with values ranging from 0 to 1, and solve equation (3) to find x_0.

For example, this calculation is used to generate an optical depth τ for the photon which has for probability:

$$P(\tau)d\tau = e^{-\tau}d\tau \tag{4}$$

Plugging this function in equation (3) gives

$$\psi(x_0) = \frac{\int_0^{\tau_0} e^{-\tau}d\tau}{\int_0^{\infty} e^{-\tau}d\tau} \tag{5}$$

which, solving for τ_0, gives

$$\tau_0 = -log(1-\xi). \tag{6}$$

The same approach also gives a way of sampling scattering angles of an isotropic distribution:

$$\mu_0 = 2\xi_1 - 1 \tag{7}$$

$$\phi_0 = 2\pi\xi_2 \tag{8}$$

with μ_0 the cosine of θ and ϕ_0 the azimuthal angle in spherical coordinates. ξ_1 and ξ_2 are two other uniform random deviates.

With the optical depth and the angles for the direction of propagation, the photons can be moved step by step, computing random values for ξ, ξ_2, and ξ_3 at each one. The photons follow a "random walk" until they leave the chosen domain or get absorbed, with a probability depending on the initial conditions of the domain. We can therefore use the Monte Carlo method to perform radiation transfer by using the method presented above and its derivatives for the initialization, the propagation, the scattering, and the absorption of the photons. The radiation transfer equation and its coefficients obtained with the Monte Carlo method are:

$$\frac{1}{c}\frac{\partial}{\partial t}I_\nu + \hat{\Omega}\cdot\nabla I_\nu + (k_{\nu,s}+k_{\nu,a})I_\nu = j_\nu + \frac{1}{4\pi}k_{\nu,s}\int_\Omega I_\nu d\Omega \tag{9}$$

with j_ν the emission coefficient, $k_{nu,s}$ the scattering opacity, and $k_{\nu,a}$ the absorption opacity. We model scattering with a similar approach by using distribution functions to obtain scattering angles, polarization angles, and changes in energy.

We usually propagate photons along straight paths in flat space, using grids which have discontinuities only at uniform cells' boundaries. As part of a previous project, we implemented a code which calculates the closest cell boundary along a photon's path in spherical coordinates. As the photon moves in the grid, calculating the closest boundary tells us which cell the photon will be in at the next scattering or absorbing event. Since the cells all have specific parameters, such as temperature or opacity, updating the cell in which the photon is present when it scatters or gets absorbed is important since these conditions influence the probabilities used in the determination of the event. However, in Athena++, we want to include the effect of strong gravity near the central massive object and therefore cannot simply move the photons in straight paths from one scattering/absorption event to the other. Spacetime is not flat in these regions and the paths followed by the photons are curved which is why a simple Monte Carlo method cannot be implemented. This thesis focuses on how to move the photons in these simulations along curved spacetime using geodesics equations.

3. METHOD

We implemented an algorithm in the Athena++ radiation MHD code[5] in order to calculate the geodesics of photons while taking general relativity into account.

The equations for a photon trajectory, obtained from the grmonty paper[4], are:

$$\frac{dx^\alpha}{d\lambda} = k^\alpha \tag{10}$$

$$\frac{dk^\alpha}{d\lambda} = -\Gamma^\alpha_{\mu\nu} k^\mu k^\nu \tag{11}$$

$$\Gamma^\alpha_{\mu\nu} = \frac{1}{2} g^{\alpha\gamma} \left(\frac{\partial g_{\gamma\mu}}{\partial x^\nu} + \frac{g_{\gamma\nu}}{\partial x^\mu} - \frac{g_{\mu\nu}}{\partial x^\gamma}\right) \tag{12}$$

with x the position of the photon, k the unit vector of its trajectory, λ the affine parameter, $g_{\alpha\beta}$ the metric and $\Gamma^\alpha_{\mu\nu}$ the Christoffel symbols of the metric. λ corresponds to the step size between each iteration of the algorithm and can therefore be set depending on the accuracy and the computational expense wanted.

$\Gamma^\alpha_{\mu\nu}$, the Christoffel symbols, are matrices which serve as metric connections and depend on the coordinate system chosen. As shown in equation (12), they are derived directly from the metric $g_{\alpha\beta}$, also a matrix, which itself depends on the coordinate system in which the calculations are performed. The metric is the primary tool used in the study of general relativity because it contains all the information about the geometry of spacetime and thus allows us to perform accurate calculations in flat or curved spacetime. It is a 4×4 matrix, one row and column for each dimension, usually denoted x, y, and z for space and t for time. Combined with the components of an infinitesimal coordinate displacement four-vectors dx^μ, the metric gives the invariant square of an infinitesimal line element or interval:

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu \tag{13}$$

This interval gives information on the nature of the geometry of spacetime. $ds^2 < 0$ means that the interval is timelike, $ds^2 > 0$ means that it is spacelike and $ds^2 = 0$ means that it is lightlike.

By choosing a coordinate system and its corresponding metric, using equation (12) to obtain the Christoffel symbols, and setting the desired stepsize λ, it is therefore possible to calculate the trajectory of a photon using equations (10) and (11) in any type of geometry or curvature.

3.1. *Verlet algorithm*

We based the implementation of the algorithm on the grmonty code[4] and the velocity Verlet algorithm given in the descriptive paper:

$$x^\alpha_{n+1} = x^\alpha_n + k^\alpha_n \Delta\lambda + \frac{1}{2}\left(\frac{dk^\alpha}{d\lambda}\right)_n (\Delta\lambda)^2 \tag{14}$$

$$k^\alpha_{n+1,p} = k^\alpha_n + \left(\frac{dk^\alpha}{d\lambda}\right)_n \Delta\lambda \tag{15}$$

$$\left(\frac{dk^\alpha}{d\lambda}\right)_{n+1} = -\Gamma^\alpha_{\mu\nu}(x_{n+1}) k^\mu_{n+1,p} k^\nu_{n+1,p} \tag{16}$$

$$k^\alpha_{n+1} = k^\alpha_n + \frac{1}{2}\left(\left(\frac{dk^\alpha}{d\lambda}\right)_n + \left(\frac{dk^\alpha}{d\lambda}\right)_{n+1}\right)(\Delta\lambda) \tag{17}$$

This algorithm uses these four equations to calculate the geodesics step by step. We chose to use the Verlet algorithm because calculating the Christoffel symbols is computationally expensive and this algorithm only requires them to be calculated once per step. For each step, the new position and the new directional vector of the photon are computed with the Verlet algorithm. We then use these new values as initial conditions for the next iteration of the loop in the algorithm which will then give the next step of the photon. With α being the index of the coordinate for which the calculation is being performed, this algorithm is therefore executed a total of four times, one per coordinate, for each step.

In this algorithm, $k_{n+1,p}$ is used as a temporary value to compute equation (16) until a final value, k_{n+1}, is calculated in equation (17). $k_{n+1,p}$ and k_{n+1} are then compared and step 2 to 4 are repeated until the difference between the two is below a chosen tolerance.

For the coordinate systems spherical-polar and cylindrical, a difficulty arises from the directional vector k. While it has the simple unit vector definition $k_x^2 + k_y^2 + k_x^2 = 1$ in cartesian coordinates, this does not hold in other systems where x, y, and z are replaced by other coordinates such as r, θ, and ϕ. For example, for spherical-polar we have:

$$(-k^t)^2 + (k^r)^2 + r^2(k^\theta)^2 + r^2 \sin^2\theta (k^\phi)^2 = 0 \tag{18}$$

Moreover, the code initializes k' while the implemented integration uses k. While this does not matter for k^t and k^r, k^θ and k^ϕ had to be corrected before being used in the algorithm with the following relations:

$$\begin{aligned} k^\theta &= \frac{k^{\theta\prime}}{r} \\ k^\phi &= \frac{k^{\phi\prime}}{r\sin\theta} \end{aligned} \tag{19}$$

With these corrections, the integration gives the right results for the calculations of the directional vectors.

4. APPLICATION

4.1. *Kerr-Schild coordinates*

In order to test our implementation in the Athena++ code, we conducted a comparison with another code, *geokerr*[6], which calculates geodesics for photons in the Kerr metric:

$$\begin{aligned} ds^2 = &-\left(1 - \frac{2r}{\Sigma}\right) dt^2 + \left(\frac{4r}{\Sigma}\right) dr\ dt + \left(1 + \frac{2r}{\Sigma}\right) dr^2 \\ &+ \Sigma d\theta^2 + \sin^2\theta \left[\Sigma + a^2\left(1 + \frac{2r}{\Sigma}\right)\sin^2\theta\right] d\phi^2 \\ &- \left(\frac{4ar\sin^2\theta}{\Sigma}\right) d\phi\ dt - 2a\left(1 + \frac{2r}{\Sigma}\right)\sin^2\theta\ d\phi\ dr \end{aligned} \tag{20}$$

The metric components and the Christoffel symbols are given in *Radiation hydrodynamics in Kerr space–time: equations without coordinate singularity at the event horizon* (Takahashi, R., 2008)[7] in Appenix A.

Along with the same metric, the Athena++ code was setup with a uniform grid composed of only one cell made bigger than the maximum radius the photon could achieved before escaping. In this cell, the opacity, the absorption, and the scattering coefficients were set to 0 in order to allow the photons to only be affected by the gravity of the black hole and therefore follow the geodesics. This way, the comparison with the *geokerr* code was made easier since the results from both codes would come purely from the computation of the geodesics using the Kerr metric. We set the black hole to have radius 2 and be non-rotating.

For the initialization of the photons in Athena++, we set the initial position to be at $r = 3.1$, $\theta = \frac{\pi}{2}$, and $\phi = 0$. With 20 photons used in the simulation, the initial directions were chosen to be $k_r = \cos\Phi$, $k_\theta = 0$, and $k_\phi = \sin\Phi$ with Φ defined as a variable to which $\frac{2\pi}{n_{photon}}$ is added for every photon initialization, where n_{photon} is the number of photons used in the simulation.

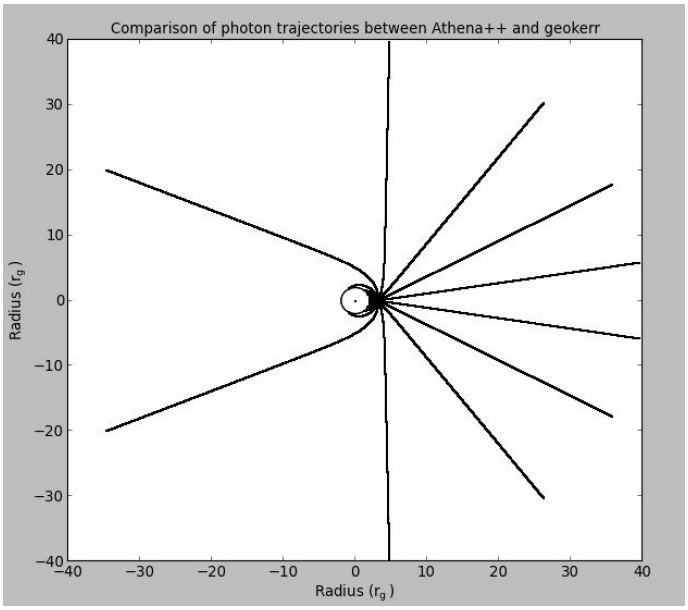

Figure 1. Comparison between Athena++ and *geokerr* with a black hole of radius of $2r_g$ centered at the origin. The photons are launched at a radius of $3.1r_g$ with different initial azimuthal directions k_ϕ. They all have $k_\theta = 0$ as part of their initial direction and therefore stay in the plane $\theta = \frac{\pi}{2}$ along their trajectories. Athena++'s output is plotted with black dots while *geokerr*'s is plotted with colored lines. Because they match closely and the step-size for Athena++ was small for better accuracy, the colored lines are hidden by the dots.

In order to test the accuracy of our implementation in Athena++, we ran the same problem on *geokerr* which uses an analytic solution for rapid and accurate calculation of null geodesics in the Kerr metric. The equations of motion from the Hamilton–Jacobi equation are reduced directly to Carlson's elliptic integrals to simplify algebraic manipulations and allow all coordinates to be computed semianalytically. In order to compare the outputs of *geokerr* and Athena++, we recorded the final position of each photon in the Athena++ code and used them as the initial position of the photons in the *geokerr* code. Starting at these positions, we took the initial direction of each photon in *geokerr* to be the negative value of the final direction of the photons in the Athena++ code. The integration was therefore done inward for *geokerr*. We then recorded the outputs from *geokerr* and plotted the results from the two codes on the same plot for comparison as shown in Figure 1.

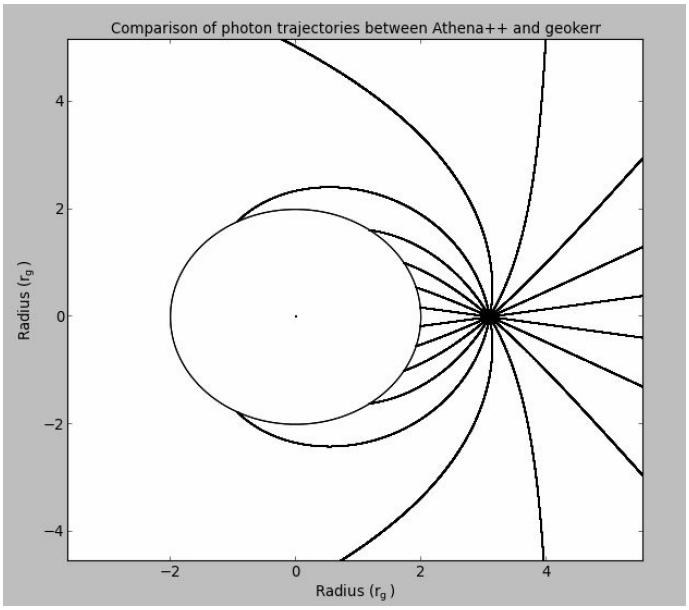

Figure 2. Figure 1 at a different scale. The effect of curved spacetime on the photons' trajectories close to the black hole appear more clearly in this figure.

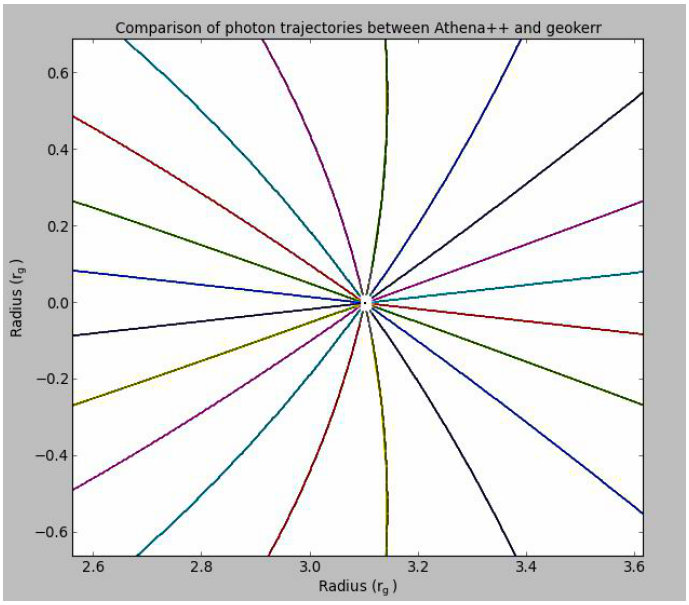

Figure 3. Figure 1 at a different scale. The colored lines showing the output from *geokerr* are visible since the scale is closer to the stepsize used in the Athena++ code.

Figures 2 and 3 present zoom-in parts of this plot, with the most zoom-in one showing the colored lines from *geokerr*.

As seen on figure 1, the photons were moved up to a maximum radius of 40 and a minimum radius of 2 which is the radius of the black hole. The step size for

Athena++ was chosen to be $\lambda = 10^{-3}$ and for *geokerr* the number of steps was set to 5000. The convergence of the two algorithm confirms the accuracy of our implementation of the geodesics integration in Athena++.

4.2. *Spherical-polar coordinates implementation*

After verifying the covariant integration in Athena++ by comparing it to *geokerr*, spherical-polar coordinates were implemented with their corresponding metric and Christoffel symbols. The metric is:

$$ds^2 = -dt^2 + dr^2 + r^2 d\theta^2 + r^2 \sin^2\theta d\phi^2 \tag{21}$$

and the Christoffel symbols are:

$$\begin{aligned}
\Gamma^r_{\theta\theta} &= -r \\
\Gamma^r_{\phi\phi} &= -r\sin\theta\sin\theta \\
\Gamma^\theta_{r\theta} &= \Gamma^\theta_{\theta r} = \frac{1}{r} \\
\Gamma^\theta_{\phi\phi} &= -\sin\theta\cos\theta \\
\Gamma^\phi_{r\phi} &= \Gamma^\phi_{\phi r} = \frac{1}{r} \\
\Gamma^\phi_{\theta\phi} &= \Gamma^\phi_{\phi\theta} = \frac{\cos\theta}{\sin\theta}
\end{aligned} \tag{22}$$

with the other elements 0.

For testing purposes, we set the spherical grid to have 4 cells in r, 1 cell in θ, and 1 cell in ϕ. While θ went from 0 to π and ϕ went from 0 to 2π, r was set to go from 5×10^{-11} to 50. Athena++ requires the minimal value of r to be different than 0 and we therefore had to set it to a very small value. The increment for the cells' size in r was 1000, meaning that the last cell was taking much of the space and the calculation was therefore performed in only one cell. The requirement on the number of cells in r comes from the code and could not be set to 1.

We compared the results of the code with the final positions as calculated from the initial position and direction using the equation of a straight line in spherical-polar coordinates. We then obtained the mean errors between the simple calculation and the outputs of the algorithm using the following set of equations:

$$\begin{gathered}
\Delta x = x_c - x_m,\ \ \Delta y = y_c - y_m,\ \ \Delta z = z_c - z_m \\
\Delta l = \sqrt{(\Delta x)^2 + (y)^2 + (\Delta z)^2}
\end{gathered} \tag{23}$$

with x_c, y_c, and z_c the calculated final position coordinates, x_m, y_m, and z_m the final position coordinates integrated using the metric, and Δl the error. We performed the calculation using 1000 photons and then averaged the errors to verify the theoretical increase in accuracy as the stepsize decreases. We present the results of this test in Figure 4.

As the step size used for the integration decreases, the difference between the final positions given by the two techniques also decreased. This is expected since the accuracy of the integration gets better as the distance between two points and their calculated positions is minimized. With a step size of $\lambda = 10^{-3}$, the average difference obtained is on the order of 10^{-4}, therefore promising a great precision with smaller step size. Figure 4 presents the trend for the mean error depending on the input stepzise.

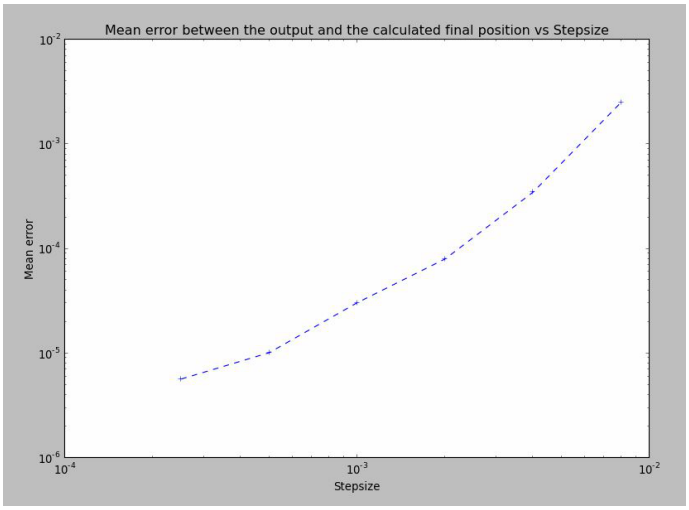

Figure 4. Mean error between the output position from the integration using the spherical-polar coordinates metric and the final position calculated versus the stepsize used.

4.3. *Cylindrical coordinates implementation*

The last implementation was in cylindrical coordinates. We used the following metric for this coordinate system:

$$ds^2 = -dt^2 + dr^2 + r^2 d\phi^2 + dz^2 \tag{24}$$

along with these Christoffel symbols:

$$\begin{aligned}
\Gamma^r_{\phi\phi} &= -r \\
\Gamma^\phi_{r\phi} &= \Gamma^\phi_{\phi r} = \frac{1}{r}
\end{aligned} \tag{25}$$

with the other elements 0.

Along with the corresponding Christoffel symbols, we used this metric to write a similar integration to the previous two in this new coordinate system.

We used a cylindrical grid with 4 cells in r, and 1 cell in both ϕ and z. The azimuth angle ϕ went from 0 to 2π, z went from 0 to 50, and r from 5×10^{-11} to 50. The number of cells in r and its minimal value were again set to respect the requirements of Athena++.

As for the spherical-polar coordinates case, we verified the accuracy of the implementation by comparing the final positions of 1000 photons as calculated by the newly implemented integration and as given by the equation of a straight line in cylindrical coordinates, both gotten when a photon would escape the grid at $r = 50$, $z = 50$ or $z = 0$. We recorded the error between these for each photon and averaged them, using Equations (23), to then plot them against the stepsize and thus show the accuracy improving as the stepsize decreases. This is shown in Figure 5.

Similar to the spherical-polar case, the accuracy decreases as the stepsize increases which is expected. With an error on the order of 10^{-4} for a stepsize of 10^{-3}, the integration promises again great accuracy for smaller stepizes.

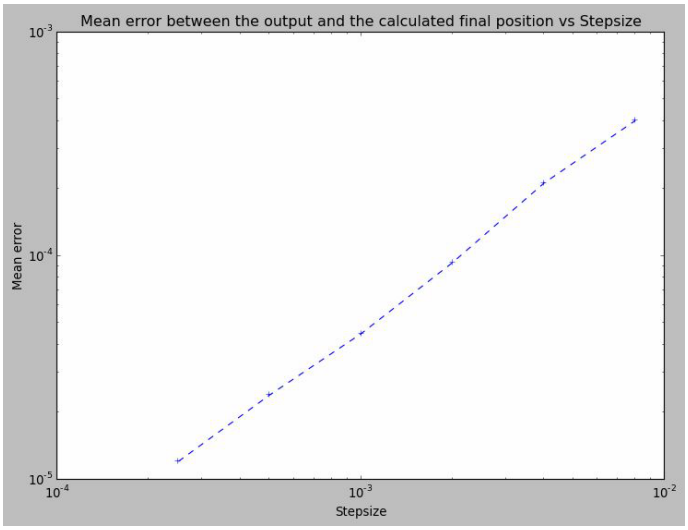

Figure 5. Mean error between the output position from the integration using the cylindrical coordinates metric and the final position calculated versus the stepsize used.

5. CONCLUSION

We implemented a covariant radiation transfer code in Athena++ which provides a way to propagate photons in curved spacetime. Using geodesics, this new code allow us to produce mock spectra while taking general relativity into account and therefore the effects of strong gravity near the black hole. This new part of the code which was implemented in Kerr-Schild, spherical-polar, and cylindrical coordinates gives us the opportunity to apply the state-of-the-art code which is Athena++ on astrophysical problems while having a radiation transfer code which is both frequency dependent and covariant. We also proved the accuracy of the implementation by comparing it to simple flat space cases for spherical-polar and cylindrical coordinates and by comparing the results for Kerr-Schild to another referenced code, *geokerr*. The next step is now for us to test more cases such as, for example, the redshifting of photons' energy as they propagate away from the black hole.

REFERENCES

[1]Kalos, Malvin H., and Paula A. Whitlock
Monte Carlo Methods
Vol. 1, John Wiley and Sons, 1986.

[2]Whitney, B. A.
Monte Carlo radiative transfer
Bull. Astr. Soc. India, 39, 2011.

[3]Kenneth Wood, Barbara Whitney, Jon Bjorkman, Michael Wolff
Introduction to Monte Carlo Radiation Transfer, 2013

[4]Joshua C. Dolence, Charles F. Gammie, Monika Moscibrodzka, Po Kin Leung
grmonty: a Monte Carlo Code for Relativistic Radiative Transport
Astrophys.J.Suppl.184:387-397,2009.

[5]White, C. J., Stone, J. M., Gammie, C. F.
An Extension of the Athena++ Code Framework for GRMHD Based on Advanced Riemann Solvers and Staggered-mesh Constrained Transport
The Astrophysical Journal Supplement, 225, 22, 2016

[6]Jason Dexter and Eric Agol
A FAST NEW PUBLIC CODE FOR COMPUTING PHOTON ORBITS IN A KERR SPACETIME
The Astrophysical Journal, Volume 696, Number 2, 2009

[7]Rohta Takahashi
Radiation hydrodynamics in Kerr space–time: equations without coordinate singularity at the event horizon
MNRAS, 383, 1155, 2008