Examining X-ray Reflection Spectra in the Galactic Centre using Monte Carlo Simulations

Michael Walls, B.Sc.

School of Physical Sciences

Dublin City University

A thesis presented for the degree of

Doctor of Philosophy

May 2017

Supervisors

Dr. Masha Chernyakova

Prof. Greg Hughes

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Masha, for her continuous support and advise over the course of my PhD, for her patience, motivation, and knowledge. I could not have imagined having a better advisor and mentor for my PhD.

Besides my supervisor, I would like to thank everyone in the DCU physics department. Although I may not have spent much time with you over the last 4 years, I do remember everyone from my undergraduate days. In particular I would like to give special mention to Enda, Tony, Ray and Dr. Laura Norci, although she has since moved on from DCU, her classes were a great inspiration in my decision to pursue Astrophysics.

My sincere thanks also go out to my colleagues in Paris, Doctors Regis Terrier and Andrea Goldwurm for all the good collaborative work we undertook together over the last 4 years.

I also want to thank my friends and family for their continued support and encouragement for the duration of my PhD. And last but not least, I want to thank my Mother, without whom none of this would have been possible.

Contents

List of Figures

List of Tables

List of Abbreviations

AGN Active Galactic Nuclei

CMZ Central Molecular Zone

GC Galactic Centre

GMC Giant Molecular Cloud

Sgr A* Sagittarius A*

Sgr B2 Sagittarius B2

VLBI Very Long Baseline Interferometry

XRN X-ray Reflection Nebula

Abstract

Examining X-ray Reflection Spectra in the Galactic Centre using Monte Carlo Simulations

-

Michael Walls

In the centre of our Galaxy lies a supermassive black hole, identified with the radio source Sagittarius A*. This black hole has an estimated mass of around 4 million solar masses. Sagittarius A* is quite dim in terms of total radiated energy, having a luminosity that is a factor of 10^{10} lower than its Eddington luminosity. However, there is now compelling evidence that this source was far brighter in the past. The evidence for which, is derived from the detection of reflected X-ray emission from the giant molecular clouds in the Galactic centre region. However, the interpretation of the reflected emission spectra cannot be done correctly without detailed modelling of the reflection process. Attempts to do so can lead to an incorrect interpretation of the data. In this thesis, I will present and outline the creation of a Monte Carlo simulation code, developed in order to fully model the complex processes involved in the emerging X-ray reflection spectra. Through the use of XSPEC table models, the simulated spectra can be compared to real data in order to derive model parameters and help to constrain the past activity of the black hole. In particular, the code is applied to observations of Sagittarius B2, the largest and most massive giant molecular cloud in the Galactic centre, and used to constrain several parameters of the cloud. This work confirms some values already present in the literature and sheds new light on one of the largest uncertainties in this field, the line of sight position. In turn, it provides some of the best estimates to date for the flaring luminosity of the incident source.

Chapter 1

Introduction

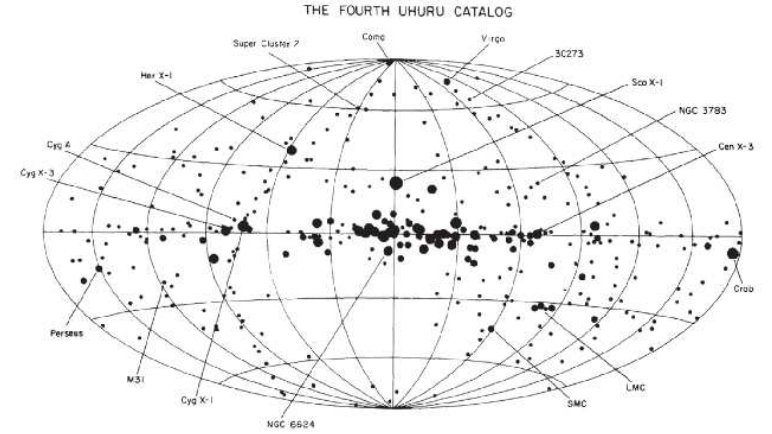
1.1 A Brief Introduction to X-ray Astronomy

It is only in the past few decades that observational astronomy has expanded to include the short wavelength high energy portion of the electromagnetic spectrum. Today we observe the universe in both the X-ray and the Gamma ray domain.

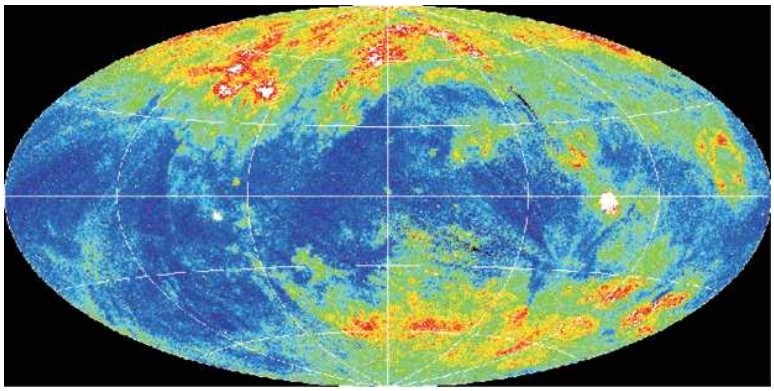
X-ray astronomy is typically thought of as covering the portion of the electromagnetic spectrum from 1.2×10^{-8} m - 2.5×10^{-12} m, or in the units of the X-ray astronomer, 0.1 keV - 500 keV. X-ray astronomy is used to observe some of the most energetic processes in the universe, from supernovae to neutron stars and black holes. The X-ray sky is filled with X-ray active galaxies (Seyfert galaxies, Quasars and other Active Galactic Nuclei (AGN)). X-ray emission is the result of cosmic sources under extreme conditions. There are several physical interactions that result in the release of cosmic X-rays. From bremsstrahlung in the million degree hot plasmas of the accretion disks around the supermassive black holes in the centre of most galaxies, to X-ray synchrotron emission produced by relativistic electrons interacting with strong magnetic fields, and Inverse Compton scattering where relativistic electrons interact with photons.

X-ray astronomy is a direct result of the space age. X-rays are unable to penetrate Earth's atmosphere and so can only be directly viewed from outside the atmosphere. The first cosmic X-ray source, Scorpius X-1, was detected in 1962 (Giacconi et al., 1962) along with the cosmic X-ray background, using a Geiger-Müller counter mounted on a NASA launched rocket. Following this discovery, numerous rocket and balloon based experiments followed, culminating in the 1970 launch of the UHURU satellite, which performed the first all sky X-ray survey,

finding 339 X-ray sources (Forman et al., 1978). Following UHURU came more advanced satellites such as ROSAT, which performed another all sky survey. Using direct imaging though the use of a Wolter telescope the ROSAT all sky survey led to the detection of almost 150,000 X-ray sources in the 0.25 - 2 keV energy range (Voges et al., 1999). Figure 1.1 shows both the UHURU and ROSAT all sky surveys (Longair, 2011).



(a)



(b)

Figure 1.1: (a) UHURU map of X-ray sources in the 2 - 6 kev energy range. (b) ROSAT X-ray map with point sources removed in the 0.25 keV soft X-ray band. Courtesy of Longair (2011).

The current generation of X-ray telescopes are far more advanced and have resolving powers far in excess of ROSAT. They are primarily, *Chandra*, *XMM-Newton*, *INTEGRAL* and *Suzaku*. These telescopes in particular have lead to an explosion of research in the field of X-ray astronomy, allowing highly resolved imaging, detailed X-ray spectroscopy and time variability studies.

The future of X-ray astronomy looks healthy as well, with the next generation of X-ray telescopes being recently deployed and more on the way. There was in particular, a very exciting X-ray telescope by the name of Astro-H/Hitomi, that was to usher in the next generation of X-ray astronomy. Having incredible resolving power Astro-H could have re-ignited the field just as *Chandra* and *XMM-Newton* did before it. In fact the work in this project had Astro-H in mind from the beginning. Unfortunately however, on March 26th 2016 there was some sort of accident on the Astro-H satellite. As of 28 April 2016, the Astro-H satellite has been declared lost and the mission a failure. There are future satellites planned such as eROSITA and the International X-ray Observatory (IXO) with planned launches in 2017 and 2021 respectively. Finally, there is the NuSTAR telescope which is a big leap forward in the field of X-ray astronomy. It was launched on 13th June 2012, and is currently performing scientific work.

1.2 The Galactic Centre

The centre of our Milky Way galaxy, hereafter referred to as the Galactic Centre (GC), is the heart of our galaxy. Unfortunately, it cannot be viewed in visible light due to the massive amount of interstellar dust between Earth and the GC, which absorbs visible light (Rieke et al., 1989). However, it can be viewed in radio, infrared and importantly X-rays, due to their ability to pass undisturbed through the interstellar dust. The GC is located approximately 8 kpc from Earth (Reid et al., 2009), making it by far the closest galactic nucleus to Earth. This means it is the best place to study not only the physical processes that occur within the centre of our own galaxy, but also within the centre of other galaxies.

The GC is a complex and dense area of the galaxy, containing a large variety of phenomena. Primarily it contains a large concentration of stars, known as the Galactic bulge. Having a bar shaped distribution, this formation dominates the gravitational potential of the GC area (Blum et al., 1995). The central few hundred parsecs of the GC contain extremely massive Molecular Clouds (MC), including several of the most massive clouds in the Galaxy (Morris and Serabyn, 1996). These are mostly contained in a region known as the Central Molecular Zone (CMZ). The CMZ is a thin layer of molecular material, containing a total mass of about $3 - 5 \times 10^7\ M_\odot$ (Dahmen et al., 1998). This molecular material makes up about 10% of the total neutral mass of the Galaxy (Morris and Ser-

abyn, 1996). Interestingly, the molecular clouds in the CMZ are quite dense with densities $> 10^4\ cm^{-3}$.

The central few parsecs of the GC contains a very dense and luminous star cluster, revealed through infrared observations (Genzel et al., 1996), along with large amounts of very hot and ionized molecular and atomic gas (Genzel et al., 1994). This gas is known as the CircumNuclear Disc (CND). Just outside the CND there is a supernova remnant known as Sgr A East, while on the other side there is a spiral structure know as Sgr A West.

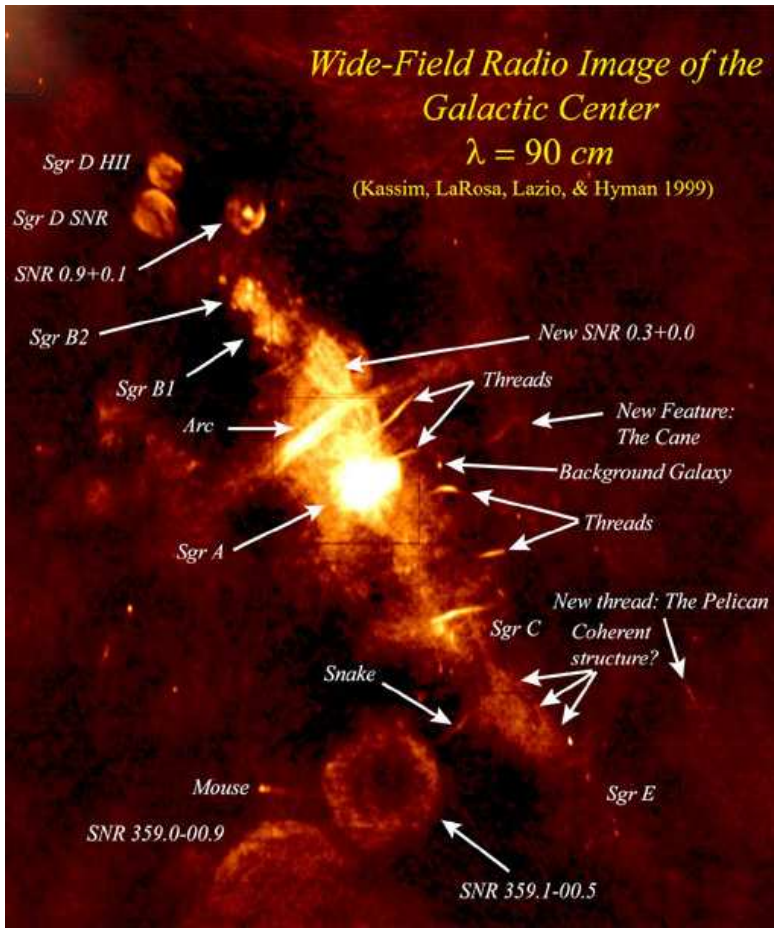


Figure 1.2: Sgr A*, the CND and the CMZ, with many of the features discussed labelled. Courtesy of Kassim et al. (1999).

These sources make up the strong radio source at the centre of the Galaxy known as Sagittarius A. Sagittarius A was first discovered in 1954 and by 1958 it had been designated as the exact centre of the Galaxy by the International Astronomical Union. At the heart of Sagittarius A there lies an extremely compact radio source known as Sagittarius A* (Sgr A*). Sgr A*, through radio observations, was found to be no more than 10 light minutes across (Shen et al., 2005).

In 2002, Schödel et al. (2002) found a star that orbits extremely close to Sgr A* and tracked its motion over several years. This, combined with later proper motion studies of the central star cluster, where several of the stars orbit with speeds in excess of 10^3 km/s (Schödel et al., 2009), has led to the conclusion that Sgr A* cannot be anything but a supermassive black hole. Gillessen et al. (2009) performed a study of the orbits of the stars orbiting the black hole and gave the most accurate mass estimation to date, with a mass of $\approx 4.31 \pm 0.38 \times 10^6\ M_{\odot}$. The likely source of the radio, and, as will be discussed in Section 1.3, the X-ray emission from Sgr A*, is an accretion disc surrounding the black hole. See Genzel et al. (2010); Ponti et al. (2013) for in depth reviews of this area. Figure 1.2 shows the Galactic centre region, specifically the CMZ. Sgr A is shown, as well as many of the massive molecular clouds that surround it. One of which is Sgr B2, a cloud that will be important to this work.

1.3 X-ray Emission from the Galactic Centre

Many X-ray telescopes over the years have observed the GC. In recent decades this monitoring has gained momentum, with long and intensive observations from *XMM-Newton*, *Chandra*, *INTEGRAL* and so on. It has become apparent that most of the X-ray emission from the GC comes from bright transient point sources, mostly X-ray binaries (Pavlinsky et al., 1994; Goldwurm et al., 1994). Nineteen point sources with a 2-10 keV peak luminosity $> 10^{34}$ erg/s have been detected in the GC area (Degenaar et al., 2012), the brightest ones being identified as accreting neutron stars or black holes. 90% of these point sources are transient, spending long durations in a quiescent state with luminosities $< 10^{33}$ erg/s and undergoing short periods of higher activity (flaring). None of these transients have been observed at luminosities much higher than 10^{37} erg/s. Furthermore, no new X-ray transients have been discovered in almost a decade, which implies that all transients with a flaring time scale less than a decade have been discovered. Sgr A* itself also has weak X-ray emission, having a luminosity of $L_{2-10} = 2 \times 10^{33}$ erg/s (Baganoff et al., 2003).

There is also diffuse X-ray emission from the GC, this diffuse emission comes from several sources. There is low temperature plasma emission (kT $\approx$ 1.0 keV) that permeates the GC. The plasma is believed to be powered by nearby supernovae (Ponti et al., 2013). Further plasma emission is also present, discovered

by Koyama et al. (1989). Having a total luminosity of $\approx 10^{38}$ erg/s concentrated in the galactic bulge, it has strong 6.7 keV Fe XXV emission lines (Revnivtsev, 2003). This emission is characteristic of a high temperature plasma (kT $\approx$ 6 kev) (Koyama et al., 2007), however the power source of this emission is still a topic of debate, see Ponti et al. (2013) for summary.

Finally, there is the diffuse 6.4 keV Fe K_α emission. Large amounts of this emission appear spatially correlated with many of the MC in the area (Sunyaev et al., 1993; Koyama et al., 1996). This led to speculation that this diffuse emission could be a reflection of past bright X-ray emission from the GC, possibly Sgr A* (Sunyaev et al., 1993).

1.3.1 X-ray Reflection from Molecular Clouds

The primary reason to believe in a past high activity period of Sgr A*, is the X-ray emission from the MCs in the surrounding area, in particular the diffuse 6.4 keV Fe K_α emission. Assuming that Sgr A* did undergo a period of higher activity and that the radiation is interacting with the surrounding molecular clouds, there are certain features and physical processes that can be expected. The MCs in the GC have hydrogen column densities on the order of 10^{22-24} cm^{-2} (Ponti et al., 2013), which equates to an optical depth of $\tau = 0.01 - 1$. X-rays will readily reflect from molecular clouds in these density ranges.

1.3.1.1 Relevant Physical Processes and Spectral Features

In this section the physical processes relevant to the reflection of X-rays from molecular clouds will be summarised. In the low energy regime ($<$ 10 keV) photoelectric absorption will heavily dominate the continuum, especially in the denser clouds. As the energy of the incident photon increases ($>$ 10 keV), Thomson/Compton scattering will begin to become the dominant process by which the photons are affected (Brown and Gould, 1970). The application of both these processes to the work undertaken in this thesis will be discussed in more detail in sections 2.2.3 & 2.2.4. An important issue of note is that of using the free electron scattering approximation. Incident photon energies of $>$ 2 keV typically allow scattering to be treated as free electron scattering given that the energy hugely exceeds the ionisation energy of hydrogen and helium (Basko et al., 1974).

However, when dealing with scattering from molecular hydrogen this is not the case. This issue is discussed further in Section 2.2.4.2

When a photon scatters it will typically lose energy, the energy loss being proportional to the energy of the incident photon and the angle of scatter. When this photon down-scattering is combined with higher density clouds, densities that lead to multiple scattering, a noticeable spectral feature will appear known as the 'Compton reflection hump'. The 'Compton reflection hump' manifests itself as 'bump' in the spectrum between approximately 20 - 100 keV. It is a result of high energy photons repeatedly down-scattering and is a tell-tale sign that multiple scattering is occurring within the source of the emission (Basko et al., 1974; Nandra and George, 1994). Scattering of this sort contains a strong angular dependence. Thomson/Compton scattering will, at low energies, follow an angular distribution of $(1 + \cos^2(\theta))$. In fact, the central tenet of this thesis is based around this angular dependence.

The spectrum will contain absorption edges, small jumps located at the point of maximum absorption, where the energy of the incident photon equals the ionisation energy of a K-shell electron. Due to the high abundance of iron, the most prominent absorption edge relevant to this project is the iron K-edge located at 7.1 keV. The spectrum will also contain emission lines. Due to the high fluorescent yield of iron, the most relevant line to this project is the iron K_α line located at 6.4 keV. This K_α line of iron is the diffuse emission discussed earlier in this section. Additionally, the scattered photons will undergo down-scattering, losing energy after the scattering interaction, which leads to what are known as Compton line shoulders. When the many reprocessed fluorescent photons proceed to be scattered after emission, they will lose some energy. This down-scattering leads to the emission line spreading out lower in energy. The size of the shoulder is related to the density and the geometry of the scattering cloud (Sunyaev and Churazov, 1996). Although they are a useful measurement tool in spectroscopy in general, the equivalent width of the iron line and the Compton shoulder are outside the scope of this project and will not be analysed.

1.3.2 Sagittarius A*

Sgr A* is confirmed to be a super massive black hole, with a mass of $4.31 \pm 0.36 \times 10^6\ M_\odot$ (Gillessen et al., 2009). The 2 - 10 keV luminosity is only $L_{2-10} = 2 \times 10^{33}$ erg/s (Baganoff et al., 2003), which is extremely dim, and is a factor of 10^{10} lower than the Eddington luminosity for a black hole of this mass. In fact no other black hole has ever been detected as having a luminosity so much lower than its Eddington luminosity. Interestingly, Sgr A* undergoes short periods of flaring activity (Baganoff et al., 2001; Goldwurm et al., 2003). X-ray flares occur at a rate of about once per day, lasting for approximately 10^{3-4} s. During these flares the X-ray intensity can increase by factors of 160 above the quiescent value (Porquet et al., 2003). In the past decade, continued observations of Sgr A* have confirmed, including in the infrared, the low luminosity quiescent state of the black hole. This situation is consistent with Sgr A* accreting stellar wind material from stars in the surrounding area (Rockefeller et al., 2004). The interesting question is whether or not Sgr A* has always been so dim, or if it undergoes periods of activity far higher than its current flaring suggests.

The evidence for this is compelling, as mentioned previously, there is the diffuse Fe K_α emission from the MC in the area. But the observations contain further evidence, such as the rapid flux drop associated with the iron K edge (Koyama et al., 1996; Murakami et al., 2000). There is also the later detection of the Fe K_β emission by Murakami et al. (2001), work which simultaneously ruled out unresolved point sources as the source of the emission. The Compton hump mentioned in Section 1.3.1.1 has been detected as well (Revnivtsev et al., 2004). These are all clear signs of X-ray reflection, and as will be explained in the following sections, Sgr A* is the likely source of the incident X-rays.

1.3.2.1 Emission Variability

The detection of variable emission from the MC in the GC, has a number of important implications. The first detection of variability in the Fe K_α emitting clouds of the GC was by Muno et al. (2007), who showed strong variability from Fe K_α emitting regions, spatially matching with molecular clouds. In fact, the detection of fast variability in the Fe K_α regions is a major reason why the Sgr A* origin theory is so well regarded, as cosmic ray interactions will not produce such variability (Dogiel et al., 2009).

Further variability was found by Koyama et al. (2008) and Inui et al. (2009), they noted a 60% decline in Fe K_α flux over 10 years from the Sgr B region. Terrier et al. (2010) found a 40% decline in the 20 - 60 keV flux from Sgr B2. They estimated an incident luminosity for the flare impacting Sgr B2 of $L_{2-10} = 1.5 - 5 \times 10^{39}$ erg/s, occurring 100^{+55}_{-25} years ago. Nobukawa et al. (2011) found a Fe K_α flux decrease by a factor of 1.9 - 2.5 from both Sgr B2 and another cloud M0.74-0.09. Notably the flux decrease from both clouds appears to be synchronised. The variability match between the MC's would appear to indicate they share a common external source (Nobukawa et al., 2011).

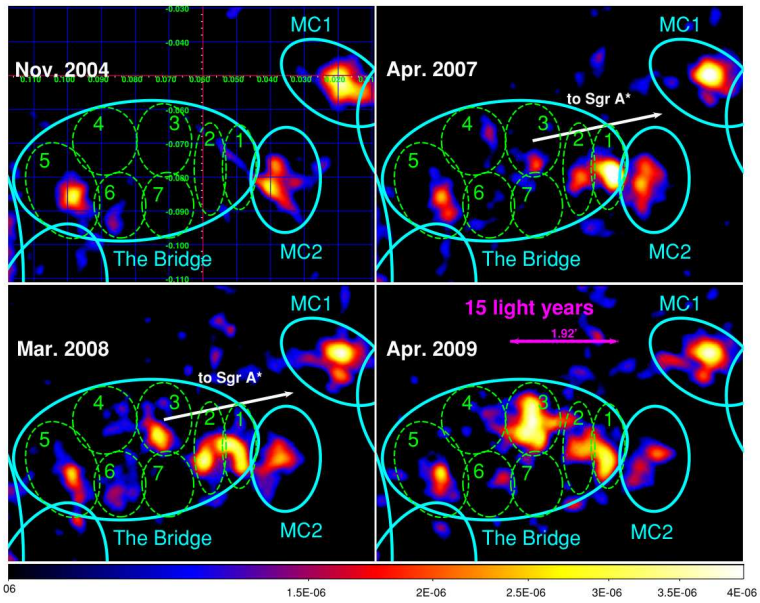


Figure 1.3: Fe K_α superluminal echo moving through molecular complex 'the bridge'. Courtesy of (Ponti et al., 2010).

In fact, variability is seen in many of the MC regions around Sgr A*, including the Sgr A and Sgr C regions (see Figure 1.2). The Sgr C region in particular was found to have large variability in *Suzaku* observations (Nakajima et al., 2009). The clump M359.38-0.00 was found to have started emitting Fe K_α photons, where before there was none, while the clump M359.43-0.07 showed shifted emission compared to earlier observations (Murakami et al., 2001). Given that these clumps appear to show different line of sight velocities (Nakajima et al., 2009), it is likely they have different line of sight distances as well. In that case, they are probably being illuminated by two distinct flares.

For the past decade the *XMM-Newton* X-ray satellite has been monitoring the 6.4 keV iron line from the CMZ. These observations led to the first detection of a

superluminal echo moving with an apparent velocity of 3c, in a molecular complex known as 'the bridge' (Ponti et al., 2010). The presence of this superluminal motion rules out an internal source for the observed emission. See Figure 1.3 which shows this superluminal light front moving through the complex (Ponti et al., 2010).

Noteworthy is the work of Cramphorn and Sunyaev (2002), who put a constraint on the luminosity of Sgr A* in the distant past, by studying the Fe K_α emission from the MC in the Galactic disc. They found very little emission which allowed them to put a constraint of $10^{-2}L_{Edd}$ for the Sgr A* luminosity in the past 10^5 years. While conversely the presence of the so called Fermi bubbles (Su et al., 2010) implies Sgr A* had a luminosity close to its Eddington limit approximately $5 - 6 \times 10^6$ years ago (Ponti et al., 2013).

Furthermore there are MCs showing rising emission as well (Capelli et al., 2011). This variation in the emission from so many MCs in the GC area heavily favours the interpretation that the MCs are reflecting a past energetic outburst of Sgr A* (Ponti et al., 2010; Capelli et al., 2012).

1.3.2.2 Morphological Evidence

There is also compelling morphological evidence that Sgr A* is the source of the reflected emission. The diffuse Fe K_α emission from Sgr B2 is shifted toward Sgr A* by ≈ 2 arcmin compared to its molecular mass distribution (Murakami et al., 2000), which clearly suggests that the source of the incident radiation is toward the GC.

Ponti et al. (2013) summarise succinctly:

> "The study of the Fe K_α emission from the MC of the CMZ shows strong indications of external irradiation, most probably due to Sgr A*. We also note that: i) spatially connected structures (such as Sgr B1 and B2 or the bridge) tend to vary in brightness with a coherent pattern; ii) molecular complexes that appear to be located close together, but display different Fe K_α light curves (such as the bridge, MC1, MC2 and G0.11-0.11; (Ponti et al., 2010)), belong to molecular complexes moving at different line-of-sight velocities. The latter point indicates that these molecular complexes are actually well separated, in three dimensions, despite being close together on the plane of the

sky. The different light curves observed from these complexes could then be tracing either different flares or different parts of the same long flare.

Moreover, we note that smaller MC generally vary on faster time-scales (with the variation fronts usually propagating close to, or higher than, the speed of light) compared to larger MC. The latter varying on a time scale being typically of the order of the cloud light crossing time. This suggests that the Fe K_α emission is induced by irradiation by an external source that is variable on both long and short time-scales. Such variability behaviour is usually observed in, for example, sources powered by accretion onto compact objects."

1.3.2.3 Possible Alternate Explanations

Although the central tenet of this thesis relies on Sgr A* being the source of the reflected X-ray emission from the GC MCs, there have in fact been other possible explanations proposed, besides Sgr A* flares, that could lead to such emission, namely cosmic rays. This has been briefly alluded to previously and will be summarised with counter points in this section.

The cosmic ray origin of most of the emission is discounted by the superluminal X-ray light front propagation (Ponti et al., 2010). However, the Arches cluster, has Fe K_α emission that is believed to be powered by a cosmic ray source (Tatischeff et al., 2012). The cluster is being impacted by a MC (Wang et al., 2006) and the computed shock is thought to provide enough power for the Fe K_α emission to be from a cosmic ray source. More recently however, (Clavel et al., 2014a) found evidence of large flux drops in the Arches cluster, implying that at least a significant fraction of the emission is from a reflected X-ray transient source.

Sufficiently bright X-ray transients located close to the MCs would theoretically be capable of producing the observed Fe K_α emission (Ponti et al., 2013). Supernovae such as the Sgr A East remnant could also be the culprit (Rockefeller et al., 2005). The shock wave from this supernova was calculated as being capable of producing the required luminosity, but it would have a very slow decay rate (Fryer et al., 2006), while the observed decay rate is more rapid (Terrier et al., 2010).

1.3.2.4 Origins of the Sgr A* Emission.

What is the physical origin of the hypothesised recent Sgr A* bright activity? As mentioned previously, the quiescent luminosity of Sgr A* is only on the order of 10^{33} erg/s. The brightest flaring event observed from Sgr A* increased the luminosity by a factor of only 160 (Porquet et al., 2003), while the luminosity required to produce the observed emission from the MCs is on the order of 10^{39} erg/s (Terrier et al., 2010). This means the required luminosity is 10^4 times more intense than the brightest observed flaring event. This would nominally suggest a difference between the process producing the observed flaring and the one that produced the past outburst (Ponti et al., 2013). Interestingly, although originally proposed to likely be a single flaring event (Ponti et al., 2010), Clavel et al. (2014b) found evidence through an in-depth analysis of X-ray emission from the Sgr A complex, that there have been multiple bright flares in the recent past.

It is possible to infer the required X-ray luminosity for the reflected emission by extrapolating from infrared observations, such as Witzel et al. (2012). Although the extrapolation requires a direct link between the X-ray and IR light curves, it still suggests that the X-ray outburst(s) was just a far more intense version of that which is producing the current flaring events.

The current flaring events are believed to be a result of Sgr A* accreting stellar wind material from nearby stars. It has been suggested by Cuadra et al. (2008) that sometimes during this process, colder clumps may fall into the region near Sgr A* with a low enough angular momentum to circularise extremely close to the black hole. The resulting accretion disc could potentially produce the required luminosity for the diffuse Fe K_α emission seen today (Cuadra et al., 2008).

Yu et al. (2011) suggested that the partial capture of a star by Sgr A* could create a shock through jet deceleration. They hypothesized that a jet would be generated by the capture of the star and would in turn be decelerated by the surrounding dense interstellar medium, resulting in the observed X-ray emission. This would also lead to the required luminosity and would reproduce the flux evolution as well (Yu et al., 2011; Terrier et al., 2010).

Zubovas and Nayakshin (2012) suggested that an asteroid passing very close to the black hole and being gravitationally disrupted could be the cause. Small objects with a radius on the order of 10 - 50 km are thought to produce a luminosity on the order of 10^{36} erg/s and are believed to possibly contribute to the

short duration smaller flaring activity observed from Sgr A* (Ponti et al., 2013). However, if a far larger planet sized object were to be gravitationally disrupted by the black hole it might produce flares with luminosities on the order of 10^{41} erg/s (Zubovas and Nayakshin, 2012).

Sazonov et al. (2012) examined the possibility of tidal interactions with stars as the source. They suggested that many unresolved late type main sequence stars in the central 0.1 pc of the GC, are tidally interacting with each other, resulting in powerful coronae and mass loss. They estimated that as a result of this, about 0.1 $M_{\odot}$ of stellar material would accrete onto Sgr A*, at a rate of a few every 10^4 years. This would produce a luminosity on the order of 10^{42-43} erg/s, while a lower luminosity could be produced by some of the torn material having a collision with another star first (Sazonov et al., 2012).

Ponti et al. (2013) believes that the accretion of interstellar material is the most probable and frequent occurrence, and is thus the most likely source of the Sgr A* outburst which gave rise to the Fe K_{α} emission; Especially given that the area around Sgr A* contains several ionised gas streams (Zhao et al., 1995) and blobs of gas and dust (Ghez et al., 2005). However, in order to more accurately determine the physical source of the emission, good estimates of the luminosity are required.

1.3.3 The Line of Sight Position.

A large point of uncertainty in luminosity estimations for the incident flare comes in the form of uncertainty in the solid angle of the reflecting cloud, as seen by the source. The area of the cloud exposed is generally unknown and unfortunately many of the MCs in the GC often overlap each other from our perspective (see Figure 1.2). However, through the use of observations such as CO surveys, the emissions of each cloud can be separated along the line of sight based on their velocity. This allows the general area of the MC to be found, though the area exposed to Sgr A* and that which is observed on earth are not necessarily the same, given the often complex shapes of MCs. However, Ponti et al. (2013) estimate this difference to be smaller than a factor of a few.

The other unknown in the solid angle estimation is the line of sight position of each cloud. Although the projected distances of the clouds are known, the actual distance between the individual MCs and Sgr A* remains unknown. It is knowl-

edge of this MC - Sgr A* distance that is an important factor in making accurate luminosity estimates, as well as flare duration and timing estimates. There is an estimated maximum divergence from the projected distance of approximately ±300 pc (Ponti et al., 2013), allowing the constraints that have been discussed already, such as $L_{2-10} \approx 10^{39}$ erg/s and the flare/flares occurring between 100 and 1000 years past. However, more accurate estimations are certainly required. There is not yet a general consensus on these line of sight positions and the goal of this thesis is to address this gap in knowledge.

Of course, the line of sight distance can be found simply by waiting. Assuming there is just the one illuminating source (Sgr A*), all the MC in the GC will eventually be illuminated by the same flare, and will reflect the same light curve. By measuring the time delay, the line of sight distance to each MC could be found. Unfortunately, finding the line of sight via waiting, would obviously take many, many years of observations. It would be far more resource efficient if the line of sight distances could be determined now and used to triangulate the position of the illuminating source and re-create the three dimensional distribution of the CMZ.

Several methods have been proposed and/or previously utilized in an attempt to calculate these line of sight distances. For example, Ryu et al. (2009) used extinction measurements of the Galactic centre plasma emission on each of the clumps in the Sgr B complex, finding the complex to lie generally on the near side of Sgr A*. Sawada et al. (2004) suggested using a comparison between 2.6mm CO emission lines and 18cm OH absorption lines, given that OH lines are expected to arise preferentially in the foreground, while the CO emission should be equal regardless of the clouds position (Sawada et al., 2004). The equivalent width of the Fe line has an angular dependence, as shown by Sunyaev and Churazov (1998), and could be used to constrain the line of sight position. This was attempted by Capelli et al. (2012). However, it is dependent on a precise knowledge of the iron abundance of the cloud being examined. It also suffers from a lack of differentiation between foreground and background positioning. In fact, all of these attempts have suffered from a lack of precision, either in specific cloud choice or cloud position.

Reid et al. (2009) made use of VLBI parallax measurements to find the distance to Sgr B2 from Earth, finding it to be $7.9^{+0.8}_{-0.7}$ kpc away, which assuming a circular orbit of the GC would place Sgr B2 $130^{+0.6}_{-0.6}$ pc in front of Sgr A*. While

helping to solve the problem of precision, the work necessitated two assumptions, that is a circular orbit of Sgr B2 and a precise knowledge of distance to Sgr A* from Earth.

The method being proposed in this thesis requires no such approximations or assumptions. It involves making use of differences in continuum shape and relative flux levels of the reflected X-ray spectra to determine the line of sight positions of the clouds. This is made possible because of the angular dependence of the scattered photons, interplaying with the geometry induced changes in absorption, effects which will be shown and discussed in Chapter 3.

1.4 Justification, Methodology and Layout of this Thesis

There have been several previous attempts to model the X-ray reflection spectra from the GC MCs and use these models to constrain relevant parameters of the reflecting MCs (though not necessarily the line of sight position). For example, the use of the `pexrav` model by Ponti et al. (2010) or the `MyTorus`[1] model by Zhang et al. (2015); Mori et al. (2015). However, these models do not reproduce precisely the spectral shape of an appropriate XRN, because they do not model the geometry of an isolated illuminated cloud.

Specifically, the `pexrav` model is a cold disk viewed from a given angle, and is only valid in Compton thick cases. It does not re-produce the iron line, and does not provide a determination of the cloud column density N_H.

Both Zhang et al. (2015) and Mori et al. (2015) have argued that the `MyTorus` model is valid and applicable. However, there are some notable shortcomings with using the `MyTorus` model, which are outlined in the appendices of both Zhang et al. (2015) and Mori et al. (2015). Primarily, `MyTorus` only deals with uniform density, has a fixed iron abundance and there is also the geometry of `MyTorus`, with scattering from the far side of the torus being an issue. This issue can be critical when the reflected spectrum is used to constrain the geometry of the reflection, namely the position of the cloud with respect to the illuminating source.

[1] http://www.mytorus.com/

To address this issue it is necessary to produce a reliable model of reflected X-rays from an isolated cloud that can be properly applied to X-ray observations (i.e. through forward folding methods). Reflection spectra for Compton thin uniform density spheres can be calculated analytically (Walls et al., 2016). Indeed, in Chapter 3 the results from these semi-analytical calculations are shown. However, these calculations are only dealing with Klein-Nishina scattering and are not taking into account binding modified multiple scattering (as such they are invalid in the Compton thick case), complex geometries or non-uniform densities.

To this end this thesis will outline the creation of a Monte Carlo code to simulate X-ray reflection spectra from molecular clouds and use it to constrain several properties of the giant molecular cloud Sgr B2, namely the dense central core column density N_H, the photon index Γ of the incident X-ray emission, the incident Sgr $A^\star$ luminosity and importantly the line of sight position of Sgr B2. It will be shown that the position of a molecular cloud has a strong influence both on the shape of the continuum and the relative strength of the 6.4 keV Fe K_α iron line. By compiling the output of the code into an XSPEC table model, model spectra can be compared to existing data and constraints placed on the location of the Sgr B2 molecular cloud.

Indeed, Sgr B2 is quite a good candidate for observing the reflected X-rays of Sgr $A^\star$. Sgr B2 is the largest, most massive and dense cloud in the CMZ (Protheroe et al., 2008). Sgr B2 has been studied as an XRN by several authors using ASCA, *Chandra*, *INTEGRAL*, *XMM-Newton*, *Suzaku* and *NuSTAR* observations (Koyama et al., 1996; Murakami et al., 2001; Revnivtsev et al., 2004; Koyama et al., 2007; Terrier et al., 2010; Ponti et al., 2013; Zhang et al., 2015) and the general interpretation is indeed one of a reflection nebula generated by a Sgr $A^\star$ outburst occurring between 100 and 300 years past. As mentioned previously Reid et al. (2009) made use of VLBI parallax measurements to find the distance to Sgr B2 from Earth, finding it to be $7.9^{+0.8}_{-0.7}$ kpc away, which assuming a circular orbit of the GC, would place Sgr B2 $\approx$ 130 pc in front of Sgr A*. While this is in nominal agreement with Ryu et al. (2009), it is dependant on a circular orbit, which as per Molinari et al. (2011); Kruijssen et al. (2015), may not be the case. Molinari et al. (2011) used *Hershel* infrared observations and subsequent velocity measurments to contruct a simple twisted ellipse model of the mass distribution around the GC. In their model, Sgr B2 lies far behind Sgr A*. While the similar but improved model of Kruijssen et al. (2015), which additionally takes into

account the orbital motion in the GC, gives a line of sight position of ≈38 pc in front of Sgr $A^{\star}$. As such, there is a lack of consensus related to the three dimensional position of Sgr B2.

Other authors have developed Monte Carlo models of the reflection process in cold molecular material, for example the early work of Sunyaev and Churazov (1998); Murakami et al. (2001). These works were subject to large approximations such as a fixed photon index and isotropic scattering as well as fixed geometry (in particular the viewing angle). The work of Odaka et al. (2011) is free from such approximations and is comparable to that presented here. However, their work concentrated more on the reflection morphology, while this work concentrates on the spectral features and the production of the XSPEC table models to allow for fitting to real observational data. Also of note is the very recent work of Molaro et al. (2016), which concentrated on the effects of clumpiness in the cloud rather than on the spectral features.

The layout of this thesis is as follows:

Chapter 2 - The Monte Carlo Code.
Chapter 2 will present the workings of the code, and simultaneously cover the physical processes being simulated.

Chapter 3 - Code Output Analysis.
Chapter 3 will present and discuss the output of the code, discussing the applicability of the code to solving the line of sight problem, and how the different parameters of the reflecting cloud will affect the output spectrum.

Chapter 4 - Data Fitting.
Chapter 4 will present a brief summary of how the data will be fit to real observational data and how the XSPEC table models are created, concluding with a summary of the observational data that has been utilized in this project.

Chapter 5 - Results.
Chapter 5 will present the results of the data fitting.

Chapter 6 - Discussion.
Chapter 6 discusses the impact of the results, the accuracy of the results, and how they fit into the broader scope of the work being done in this area.

Chapter 7 - Conclusions and Future Prospects.
Chapter 7 will present conclusions and a look at future work.

Chapter 2

The Monte Carlo Code

In this chapter the Monte Carlo code will be presented, beginning with a brief summary of the Monte Carlo approach and its scientific applications. Then the general design of the code and the various physical processes that are being simulated will be discussed, followed by details of how each physical process is implemented in the code. Finally, relevant specifics of the code implementation will be covered.

2.1 The Monte Carlo Method

The Monte Carlo method is a computational method that makes use of the repeated sampling of random numbers in order to generate numerical results; results that are very difficult or indeed impossible to obtain through a purely mathematical approach. Put another way, it approximates solutions to quantitative problems through statistical sampling. It is often used to solve physical and mathematical problems. A very early form of the Monte Carlo method can be seen in the well known Buffon's needle experiment from the 18th century, where the value of π is estimated by dropping needles in a random pattern on the floor. More recently, the modern Monte Carlo method was first conceived in the 1940s by **Stanislaw Ulam** whilst working at the Los Alamos National Laboratory. He gives the following recount of his inspiration (Eckhardt, 1987).

> *"The first thoughts and attempts I made to practice [the Monte Carlo Method] were suggested by a question which occurred to me in 1946 as I was convalescing from an illness and playing solitaires. The question was what are the chances that a Canfield solitaire laid out*

> *with 52 cards will come out successfully? After spending a lot of time trying to estimate them by pure combinatorial calculations, I wondered whether a more practical method than "abstract thinking" might not be to lay it out say one hundred times and simply observe and count the number of successful plays. This was already possible to envisage with the beginning of the new era of fast computers, and I immediately thought of problems of neutron diffusion and other questions of mathematical physics, and more generally how to change processes described by certain differential equations into an equivalent form interpretable as a succession of random operations. Later [in 1946], I described the idea to John von Neumann, and we began to plan actual calculations."*

Von Neumann, Ulam and their colleague Nicholas Metropolis were the inventors of the name Monte Carlo method, which they named after the city of Monte Carlo in Monaco. Nicholas Metropolis and Stanislaw Ulam published the first article detailing the method in 1949 (Metropolis and Ulam, 1949). It was the need for random numbers when utilizing the Monte Carlo method that led to the creation of pseudo-random number generators (see Section 2.2.1.1). The Monte Carlo method has many possible applications including the resolution of physical problems ranging from fluid dynamics to weather forecasting. The Monte Carlo method is quite often applied to mathematical problems as well, where one of the largest uses is in numerical integration, when the curse of dimensionality[1] causes standard numerical integration to fail.

A typical Monte Carlo simulation follows a standard pattern:

- Define a list of possible inputs
- Generate all inputs randomly from some distribution
- Perform deterministic computation on the inputs
- Collect results

There are two important conditions to consider: the random inputs must be uniformly distributed over the distribution and there should be a very large number of inputs. Without satisfying these conditions the resulting simulation will be poor. The code presented in this thesis follows this standard pattern.

[1] https://en.wikipedia.org/wiki/Curse_of_dimensionality

As discussed in Chapter 1, the primary motivation for this project is to correctly simulate the X-ray reflection spectra of giant molecular clouds in the Galactic centre. X-ray reflection spectra are a perfect candidate for Monte Carlo simulations, as all the physical processes that determine the form of such spectra are probabilistic in nature and are well understood. It is possible to calculate a simple reflection spectra using analytical methods (Walls et al., 2016) and this is discussed further in Chapter 3. However, there are a few areas in which the analytical approach will fail, such as, complex geometry, non-uniform density and multiple scattering. These areas are all covered by the Monte Carlo code, as will be shown in the following sections.

2.2 Code Design

The code itself is written in C++, specifically C++11. It was compiled using g++ 4.3.4. Compilation was forced to use the flag 'c++0x' rather than 'c++11' as the compiler version available on the ICHEC system (g++ 4.3.4) only supported 'c++0x'. This had no bearing on the final results, as the C++11 features used are negligible and fully included in c++0x.

The code follows the life of a single photon from generation to observation, all in a fully realised 3D environment. There is an absolute coordinate system, which tracks the photon's position. Every position has an X, Y, and Z coordinate with units in cm. The source is located at (0, 0, 0). The cloud centre is kept at a Z value of 0 in order to keep it located in the same plane as the source and the observer, which represents the Galactic plane. The observer is located at $(2.6 \times 10^{21}, 0, 0)$, which is equivalent to 840 parsecs. This is a factor of 10 lower than the actual distance of Earth to the Galactic centre, but was necessary to achieve good statistics and has no effect on the final result. Each photon is generated at the source (0, 0, 0).

The code is quite straight forward in its basic operation, and follows an approach somehwat similar to Leahy and Creighton (1993) from which it is inspired. The photon is first generated (Section 2.2.2), then it enters the cloud. Once within the cloud the photon has a chance to interact with the atoms and molecules of the cloud through photoelectric absorption (Section 2.2.3) and scattering (Section 2.2.4). If the photon is absorbed it has a chance to re-emit as a fluorescent photon (Section 2.2.3.1). After the photon escapes the cloud it is binned (Section

2.2.8). The basic operation of the code can be seen as a flow chart in Figure 2.1. Typically any one code run will consist of 10^9 input photons.

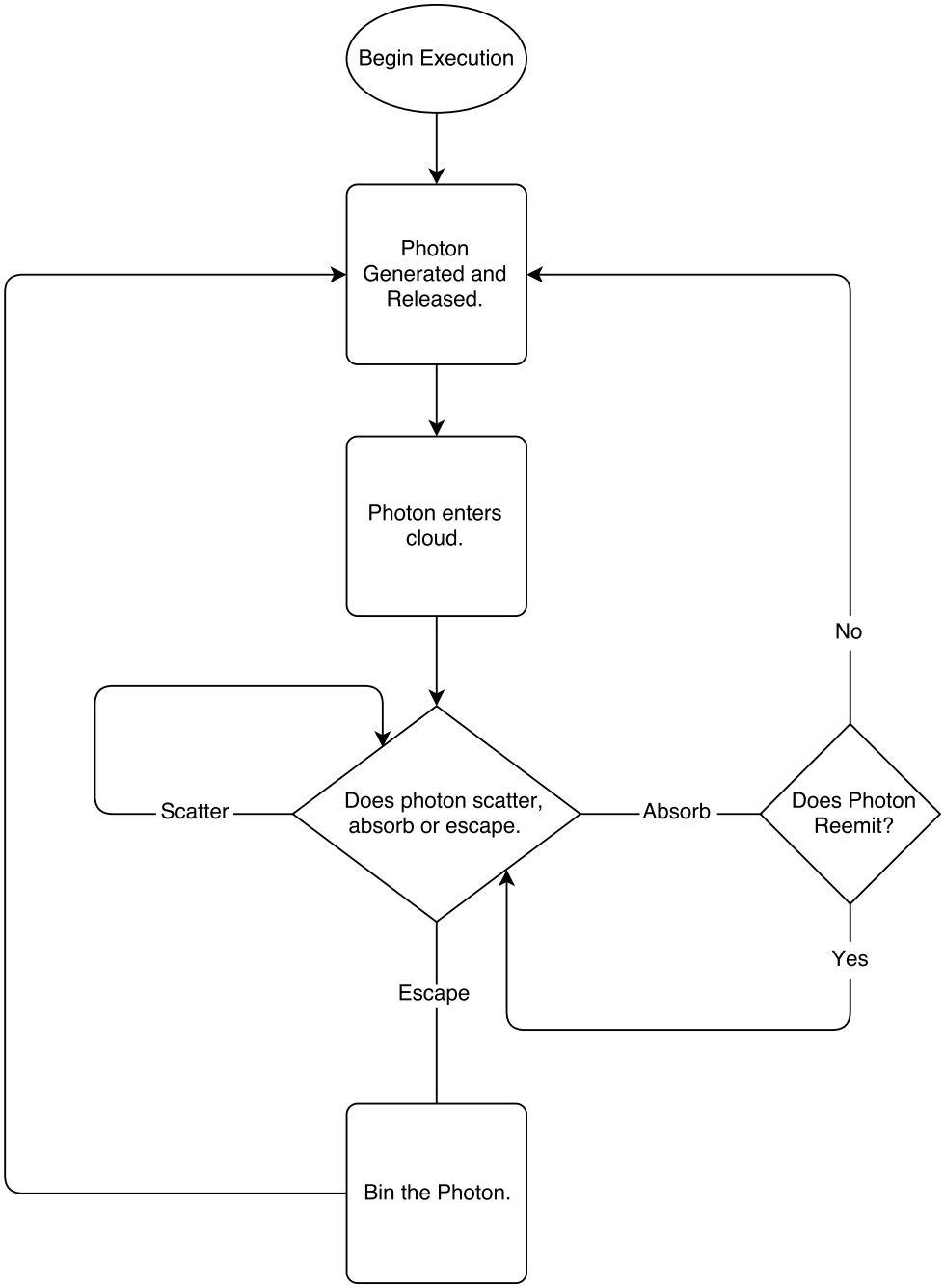


Figure 2.1: Basic flowchart for the Monte Carlo code.

As discussed in Chapter 1, the primary goal is to establish the line of sight angular position of the cloud relative to Earth. As such, the cloud's position is variable along the line of sight. See Figure 2.2, where the Galactic plane is illustrated as seen from above and the line of sight angle of the cloud is shown varying, as in the simulation. It can be seen that as the angle θ varies, the angle through which photons will have to scatter in order to reach the observer will also have to vary. For example, if the cloud is directly between the source and the observer, e.g. at an angle of 0 degrees, any photons observed (excluding multiple scatters) will have been scattered through very small angles or indeed not scattered at all. Conversely, if the cloud is behind the source relative to the observer at an angle of 150 degrees, photons that do not scatter at all will not be observed, and any photons that do scatter will have to scatter through a far larger angle in order to reach the observer.

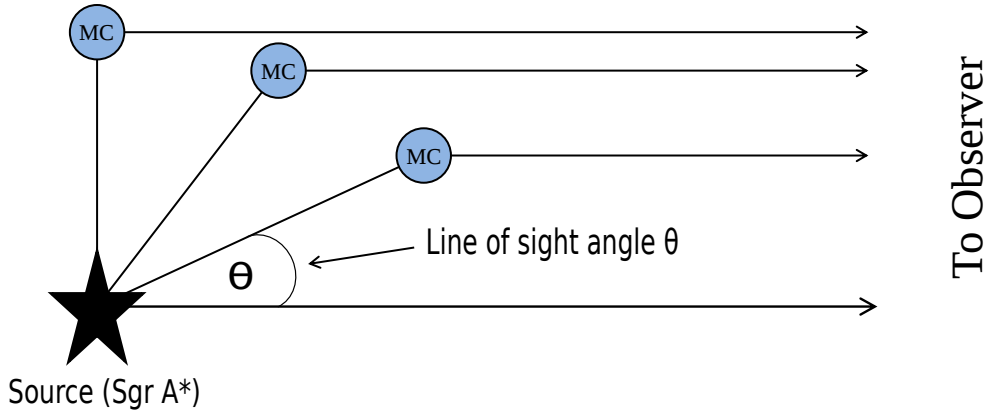


Figure 2.2: Geometry of the simulated system composed of an illuminating source (star), a reflecting cloud (MC), and an observer at infinity. Looking down into the plane of the galaxy, the line of sight angle theta (θ) is shown.

The code takes several input parameters:

- Photon index of the input spectrum
- Geometry of the cloud
- Density of the cloud in units of $\frac{atom}{cm^3}$
- Density profile of the cloud
- Radius of the cloud in units of cm

It is by changing these various input parameters, along with the line of sight position, that will lead to markedly different output spectra. The geometry of the cloud is decided before compilation and the line of sight parameter is introduced upon spectral extraction (Section 2.2.8).

The Monte Carlo method requires the random sampling of probabilistic distributions, sampling methods are covered in Section 2.2.1.

After a photon is generated, the code determines which interaction the photon undergoes (absorption or scattering) by way of cross section comparison. The cross sections for each process are well known; how each cross section value is determined is covered in sections 2.2.3 and 2.2.4 respectively. It is also possible for the photon to escape the cloud without undergoing any interaction, these photons that escape directly are not binned. This results in the 0 degree case being un-physical, a fact which is discussed further in Chapter 3

The geometry of the cloud can also be modified, however only simple geometrical shapes are modelled thus far. Currently spherical and flat-capped cylinder functions have been written, but other shapes can be added at a later date. The distance from the photon's current location to the edge of the cloud is important in determining the fate of the photon (absorb/scatter/escape). This distance is found by means of a ray-tracing method which is covered in Section 2.2.6 along with the geometry of the cloud. Non-uniform density profiles are simulated using three density profiles, r^{-2}, e^{-r} and Gaussian (Section 2.2.7.1).

In Section 2.2.5 the use of the **xraylib** library is discussed. Finally, photon binning and data output are covered in Section 2.2.8.

The code can be seen in full in Appendix B.

2.2.1 Sampling

As mentioned in Section 2.1, the Monte Carlo method involves the random sampling of known distributions. The two main sampling methods used are the inverse transform sampling method and the rejection sampling method. This code makes use of both methods and they are covered in sections 2.2.1.2 and 2.2.1.3 respectively. In order to do this random sampling, the production of random numbers is required. Specifically, a random number between 0 and 1, known hereafter as a Uniform Random Deviate (URD) which is covered in Section 2.2.1.1.

2.2.1.1 Uniform Random Deviate

A uniform random deviate is a random number that lies within a specific range, which for the purposes of this project is typically the range (0, 1). The generation of a random deviate is necessary for the implementation of the Monte Carlo method as mentioned in Section 2.1. The generation of URDs is done using a pseudo-random number generator (PRNG). A PRNG is an algorithm that generates deterministic numbers in a seemingly random distribution. The code makes use of the classic C function *rand()*, which is included in the **cstdlib** standard header. Although this is a very old algorithm and there a far superior algorithms available for PRNGs today, the high volume of photons that the code is processing means that *rand()* is more than satisfactory in this case. This was tested by comparing code output generated using *rand()* to code that made use of the C++11 **random** headers mersenne twister algorithm[1]. There were no differences in code output between the two.

rand() generates random numbers between the values 0 and RAND_MAX. The value of RAND_MAX is library specific, but is always greater or equal to 32767, which is the maximum value of a signed 16-bit integer. In order to generate URDs in the range (0, 1), *rand()* returns some factor of RAND_MAX divided by RAND_MAX, which results in returning a number between 0 and 1.

[1] http://en.cppreference.com/w/cpp/numeric/random/mersenne_twister_engine

2.2.1.2 Inverse Transform Sampling

The inverse sampling method works by randomly sampling the inverse of the cumulative distribution function of a probability density function. Typically, and in the case of this code the following procedure is followed. The intended goal is to randomly sample values x from a distribution function $f(x)$. The function $f(x)$ must thus be normalised to become a probability density function $p(x)$ by integrating between the min and max values, equating to 1, and solving for N, see equation (2.1).

$$p(x) = N \int_{min}^{max} f(x) = 1, \tag{2.1}$$

where $-\infty < min < max < +\infty$.

The cumulative distribution function $cdf(x)$ of this probability density function $p(x)$ must then be found by integrating between the lower limit and x, see equation (2.2)

$$cdf(x) = \int_{min}^{x} p(x) \tag{2.2}$$

The cumulative distribution function is the probability that a random variable will evaluate to less than or equal to x. Thus the $cdf(x)$ evaluates to a number between 0 and 1, therefore it can be equated to a URD between 0 and 1 and inverted to sample random values x, see equation (2.3).

$$x = cdf^{-1}(\tau), \tag{2.3}$$

where τ is a URD between 0 and 1.

2.2.1.3 Rejection Sampling

In order to perform inverse transform sampling, the function to be sampled must be integrable and the evaluating integral must not be computationally expensive. If this is not the case, rejection sampling may be used. Rejection sampling, sometimes called the acceptance-rejection method, or hit and miss, is a method used to generate random observations that follow a particular distribution. Simply put, completely random values within a certain range are generated, every value that is 'under the curve' or satisfies the distribution at that point is accepted,

and those that do not are rejected. With this method it is possible to re-create the distribution desired using random sampling.

If there is some distribution $f(x)$ that needs to be randomly sampled, another known distribution $p(x)$ is taken. This distribution $p(x)$ must entirely encompass the distribution $f(x)$ and be easily evaluated computationally. The simplest form of the distribution $p(x)$ is to encompass the entire range of values over which $f(x)$ can be evaluated. Such as $p(x) = max(f(x))$, $p(x)$ is simply the maximum value of $f(x)$. If $f(x)$ is sharply peaked this can lead to a lot of rejected samples, however for distributions $f(x)$ that are not sharply peaked, it is the easiest and simplest method, and is the method employed in the code. An example can be seen in Figure 2.3[1] where the blue points are rejected samples and the red are accepted samples, with the red samples accurately re-creating the distribution $f(x)$.

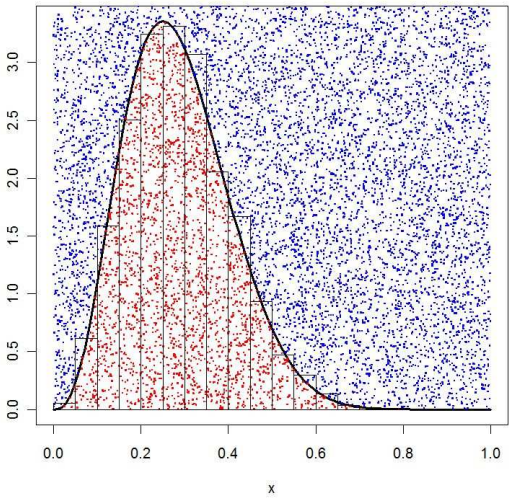


Figure 2.3: Rejection sampling example.

In the code a random value $x = B \times i$, where i is a URD, is generated in the range $0 < x < B$. B is the maximum value of x over which the distribution $f(x)$ is to be sampled. $f(x)/max(f(x))$ will return a value in the range $0 < f(x)/max(f(x)) < 1$ for any input x. $f(x)/max(f(x))$ is then compared to a URD, if the URD is less than $f(x)/max(f(x))$ then the value of x is accepted, otherwise it is rejected and a new sampling is attempted.

[1] https://www.lancaster.ac.uk/pg/jamest/Group/images/reject.jpg

2.2.2 Photon Generation and Movement

The source emits its photons sequentially in a power law distribution in the energy range E_{min} to E_{max}. The probability density function for photon production that produces this power law is given by $P(E) = NE^{-\Gamma}$, where Γ is the photon index. As per the procedure discussed in Section 2.2.1.2 it must be solved for the normalisation constant N, see equation (2.4).

$$N = \begin{cases} \frac{1-\Gamma}{E_{max}^{1-\Gamma} - E_{min}^{1-\Gamma}}, & \Gamma \neq 1 \\ \\ \frac{1}{\ln(E_{max}/E_{min})}, & \Gamma = 1 \end{cases} \tag{2.4}$$

In this project E_{min} and E_{max} are set to 1.0 and 400 keV respectively, unless stated otherwise. The cumulative distribution function must then be found and inverted. Thus, if τ is a uniform random deviate between 0 and 1, the energy of the generated photon is given by equation (2.5).

$$E = \begin{cases} \sqrt[1-\Gamma]{[E_{max}^{1-\Gamma} - E_{min}^{1-\Gamma}]\tau + E_{min}^{1-\Gamma}}, & \Gamma \neq 1 \\ \\ E_{min}(E_{max}/E_{min})^{\tau}, & \Gamma = 1 \end{cases} \tag{2.5}$$

Each photon has a position (x, y, z) and a direction of motion (θ, ϕ), where θ is the azimuthal angle and ϕ is the polar angle in the absolute coordinate system. The photon is released with a direction of motion that is uniform within the solid angle of the cloud as seen by the source. As the photon moves, its position is updated, with the new position being given by equation (2.6).

$$\begin{aligned} x &= x_0 + sa, \quad a = \sin\phi\cos\theta \\ y &= y_0 + sb, \quad b = \sin\phi\sin\theta \\ z &= z_0 + sc, \quad c = \cos\phi, \end{aligned} \tag{2.6}$$

where s is the distance the photon moves between interactions.

In order to determine s, the distance the photon moves, the column density the photon moves through must be calculated. Where the column density is the total number of atoms per unit area, as measured along a column; that the photon will 'see' before undergoing its next interaction. The probability distribution function for photon attenuation is given by equation (2.7).

$$P(N) = \sigma e^{-\sigma N}, \tag{2.7}$$

where N is the column density traversed and σ is the absorption or scattering cross section. Following the procedure in Section 2.2.1.2 once again, when τ is a uniform random deviate between 0 and 1, the column density N, encountered before an interaction, is given by equation (2.8).

$$\begin{aligned} N_a &= \frac{-\ln(\tau)}{\sigma_a} \\ N_s &= \frac{-\ln(\tau)}{\sigma_s}, \end{aligned} \tag{2.8}$$

where σ_a and σ_s are the absorption and scattering cross sections respectively and N_a and N_s are the absorption and scattering column densities encountered before interaction respectively. At every point of interaction, the column density N_e, between the photons position and the edge of the cloud along the photons direction of movement is also calculated, by means equation (2.9).

$$N_e = \rho \times L, \tag{2.9}$$

where N_e is the column density, ρ is the density of the cloud and L is the distance to the far side of the cloud along the photon's direction of motion. This distance is calculated by the ray-tracing method covered in Section 2.2.6. The lowest column density of these three N_a, N_s and N_e is then used to determine if the photon was absorbed, scattered or escapes from the cloud. If N_e is the lowest column density, then the photon is considered to have escaped the cloud along its current trajectory. If either N_a or N_s is the lowest column density, then the distance s the photon travels between its previous position and the position of its new interaction, is given by equation (2.10).

$$s = \frac{N}{\rho}, \tag{2.10}$$

where N is either the absorption or scattering column density, and ρ is the density of the cloud. The photon is then propagated this distance using equation (2.6).

2.2.3 Absorption

Photoelectric absorption is the process by which an incident photon gives all of its energy to an inner shell electron of an atom. The photon is destroyed in the process. If the energy of the incident photon is at or above the binding energy of the electron, the electron will be ejected from the atom as a photo-electron. It is also possible that after an electron is ejected, another electron from a higher shell will drop down to fill the missing hole of the ejected electron in a process known as fluorescence (covered in Section 2.2.3.1). Photoelectric absorption is also known as the photoelectric effect. However, because this work is more interested in the absorbed photon than the released electron, it is called photoelectric absorption.

The probability of absorption occurring is at its greatest when the incident photon is equal to or greater than the binding energy of the electron, and when said electron is an inner shell electron (e.g. K-shell). This is what leads to absorption edges, as once the energy of the incident photon reaches the binding energy of the K-shell electrons, the chance of absorption increases dramatically.

All of the absorption cross sections used in the code are sourced from the NIST XCOM database[1], which sources its data from a number of published works, references to which are available[2]. The XCOM database provides cross sections for all elements with $Z \leq 100$, at energies from 1 keV to 100 GeV. The strength of the total absorption cross section is dependent on the contribution from each element that is being considered, and scaled based on the relative abundance of that element. Typically, in this project, solar abundances will be used. Solar abundances are the relative abundance of each element in the Sun. The values of these solar abundances are shown in Table 2.1 (Lodders, 2003), the number being $\log_{10}$ [N(element)/N(H)] + 12.

Other elements are rare in Giant Molecular Clouds (GMC) and are not considered. The total absorption cross section σ_{abs} is thus a sum of the photoelectric absorption cross section for each element scaled by its relative abundance. The total absorption cross section is given by equation (2.11).

$$\sigma_{abs} = \sum_{H}^{Ni} \sigma_i \times [N(element)/N(H)] + 12, \tag{2.11}$$

[1] http://www.nist.gov/pml/data/xcom/

[2] http://physics.nist.gov/PhysRefData/Xcom/Text/ref.html

H	12	Al	6.51
He	11	Si	7.64
C	8.66	S	7.23
N	7.89	Ar	6.57
O	8.91	Ca	6.34
Ne	8.05	Fe	7.59
Na	6.31	Ni	6.27
Mg	7.62		

Table 2.1: Solar abundances, the abundance being $\log_{10}$ [N(element)/N(H)] + 12, where each value in the table is N(element).

where the summation from hydrogen to nickel does not take into account all elements, only those shown in Table 2.1. The total photoelectric absorption cross section as calculated by equation (2.11) can be seen in Figure 2.4, along with two examples of the input cross section values for hydrogen and iron, these represent the two most important elements that are of interest in this project.

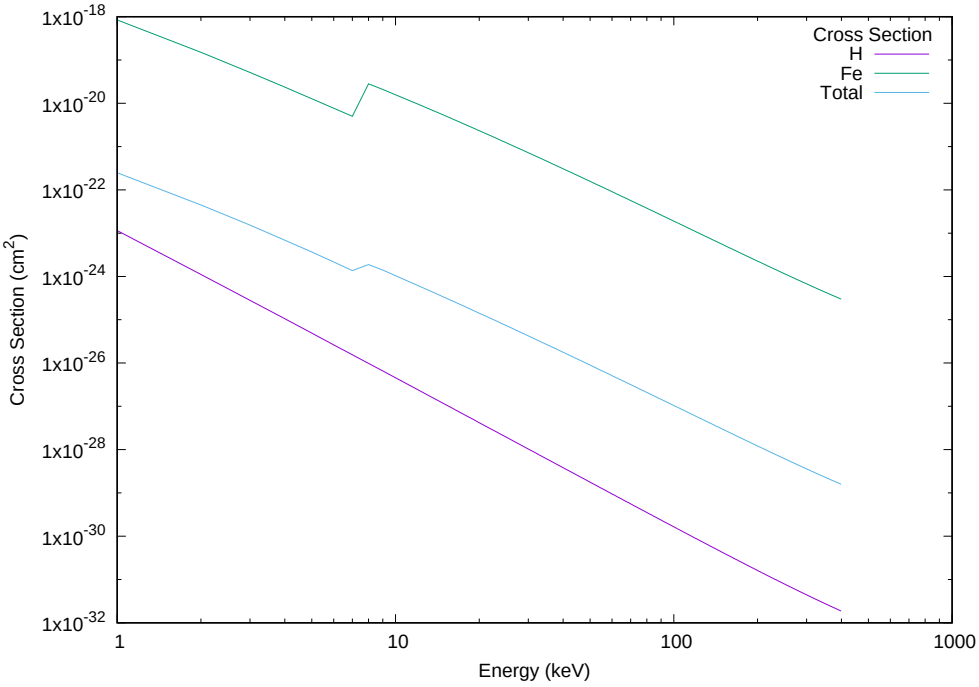


Figure 2.4: Photoelectric absorption cross sections for hydrogen, iron and the total absorption cross section.

Note the strong absorption edge that is present at 7.1 keV in the iron cross section. Similarly, note the relative strength of the iron cross section compared

to Hydrogen, iron is a much better absorber than hydrogen and this is reflected in the cross section values. In fact, the absorption cross section is approximately proportional to the atomic number of the absorber as shown by equation (2.12).

$$\sigma_{abs} \propto \frac{Z^5}{E^3}, \tag{2.12}$$

where Z is the atomic number of the interested absorber and E is the energy of the incident photon. The cross section value is calculated with the addition of the *barn* factor, which is for unit conversion. The data from the XCOM database comes in units of barn/atom, where *barn* has a value of 1×10^{-24} cm^2.

2.2.3.1 Fluorescence

As mentioned briefly in Section 2.2.3, when an incident photon is absorbed by an inner K-shell electron of an atom, the K-shell electron will get released from the atom as a photo-electron. As a result a higher shell electron will drop down to fill the hole created. As this electron drops down it must lose energy, specifically the energy difference between the two shells. This energy can often be released in the form of a new photon, a photon which will have a very specific energy, namely the energy lost by the electron that filled the hole. This effect is what leads to emission lines. It is also possible for this energy to be transferred to another electron in the atom, and for this electron to be released as well. This is known as the Auger effect. However, as the Auger effect does not lead to a photon being released it is not of any further importance. The probability of a photon being released is known as the fluorescence yield.

In this project specifically, there is one important emission line in particular, which is the 6.4 keV K_α line of iron. This line is very prominent in the energy ranges being examined. There is also the less prominent 7.1 keV K_β line. In order to determine if an absorbed photon leads to a re-emitted photon, the code goes through the following process.

If the photon is determined to have been absorbed, there is a chance that it will be re-emitted by an iron atom. The probability that it was absorbed by iron is taken from the abundances mentioned in Section 2.2.3. The fluorescence yield of iron is taken to be 0.34 (Bambynek et al., 1972). This means that for every photon that exceeds the K-shell binding energy of iron, which is then absorbed by an iron atom, there is a 34% chance that a new photon will be released. The

binding energy of K-shell electrons in iron is taken to 7.114 keV (Bearden and Burr, 1967). The probability of K_β fluorescence relative to that of K_α fluorescence is taken to be 0.13 (Kaastra and Mewe, 1993). There are actually two emission lines of iron around 6.4 keV, they are $K_{\alpha 1}$ and $K_{\alpha 2}$. However, given the low resolution of current instruments, they are treated as one single line. The energy of K_α fluorescence is taken to be 6.399 keV and that of K_β is taken to be 7.085 keV.

A photon can thus be said to have been re-emitted, if a URD satisfies equation (2.13).

$$\tau \leq \frac{\sigma_{Fe}(E) \times \frac{1}{10^{H-Fe}}}{\sigma_{abs}(E)} \times 0.34, \tag{2.13}$$

where τ is a URD, $\sigma_{Fe}(E)$ and $\sigma_{abs}(E)$ are the iron and total absorption cross sections at the energy of the incident photon respectively, H and Fe are the Hydrogen and Iron abundances (as shown in Table 2.1), and 0.34 is the iron fluorescence yield.

If the code determines that a photon has been re-emitted, it is then randomly decided if it is either a K_α or K_β photon based on the relative probability 0.13. Finally it is given a new direction of propagation which is entirely isotropic.

2.2.4 Scattering

Scattering is the process by which incident photons will interact with the electrons of an atom. This interaction results in a change of direction for the incident photon and potentially an energy change as well. Scattering by elements other than hydrogen and helium is negligible in molecular clouds, so all metals are ignored.

It was initially thought that the binding energy of electrons in hydrogen and helium was so low compared to the energy of the incident photons that the code could treat the electrons as if they were unbound. Treating the electrons as if they are unbound simplifies the process of simulating scattering. However, it became apparent in the later stages of this work that when dealing with scattering from molecular hydrogen, the effects caused by the electrons being bound is not at all insignificant and cannot be ignored. As will be discussed in Chapter 3, the comparisons with the analytical calculations required the use of the unbound

approximation, thus both bound and unbound scattering will be covered in this section.

2.2.4.1 Unbound Scattering

Unbound scattering in this context is the scattering of photons by free electrons. The classical approximation of this scattering is known as Thomson scattering. The Thomson differential scattering cross section is given by equation (2.14) (Longair, 2011) and can be seen in Figure 2.5. The angular dependence of Thomson scattering is evident from Figure 2.5, where forward and back scattering are equally as likely. However, scattering at an angle of 90° is half as likely.

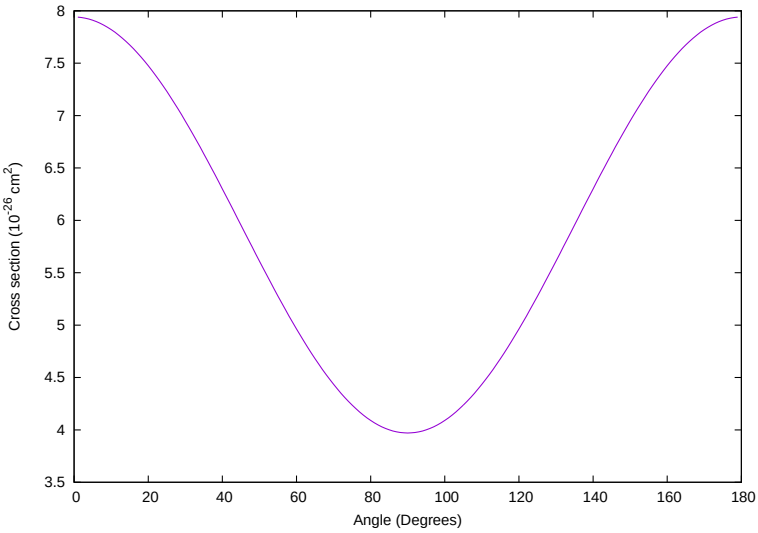


Figure 2.5: Thomson differential scattering cross section

$$\frac{d\sigma_{Th}}{d\Omega} = \frac{1}{2}r_e^2(1 + cos^2\theta), \tag{2.14}$$

where $r_e = \frac{e^2}{m_e c^2}$ is the classical electron radius, m_e is the mass of the electron, and c the speed of light in vacuum. Integrated over all angles to give the total Thomson scattering cross section, equation (2.14) yields equation (2.15).

$$\sigma_{Th} = \frac{8\pi}{3}r_e^2 = 6.65 \times 10^{-25} cm^2, \tag{2.15}$$

where again r_e is the classical electron radius. Thomson scattering assumes that the scattering interaction is elastic, that is to say the energy of the incident photon does not change as a result of the interaction. However, as the energy of the incident photon increases this is no longer a valid approximation. The

scattering becomes inelastic (there is an energy change in the scattered photon) and it becomes known as Compton scattering. Thus, Thomson scattering is actually just a low energy approximation of Compton scattering. Compton scattering on free electrons is described by the Klein-Nishina formula (Klein and Nishina, 1929). The code calculates the scattering cross section directly from the Klein-Nishina formula for all interested energy levels (1 - 400 keV), which includes those energy levels that could be accurately approximated by Thomson scattering (1 - 50 keV). The energy change that an incident photon undergoes as a result of Compton scattering from a non-relativistic electron is given by equation (2.16) (Longair, 2011).

$$\epsilon\prime = \frac{\epsilon}{1 + \frac{\epsilon}{m_e c^2}(1 - \cos\theta)}, \tag{2.16}$$

where ϵ and $\epsilon\prime$ are the photon energy before and after scattering respectively. θ is the angle through which the photon has scattered. The differential scattering cross section for Compton scattering as given by the Klein-Nishina formula is given by equation (2.17) (Rybicki and Lightman, 1985) and can be seen in Figure 2.6. Evident from Figure 2.6 is the fact that at low energy (1 keV) Compton scattering is accurately approximated by Thomson scattering (compare with Figure 2.5). However, as the energy of the incident photon increases the Klein-Nishina differential scattering cross section begins to heavily favour forward scattering.

$$\frac{d\sigma_{KN}}{d\Omega} = \frac{r_e^2}{2} \times \frac{\epsilon_\prime^2}{\epsilon^2} \times \left(\frac{\epsilon}{\epsilon\prime} + \frac{\epsilon\prime}{\epsilon} - sin^2\theta\right). \tag{2.17}$$

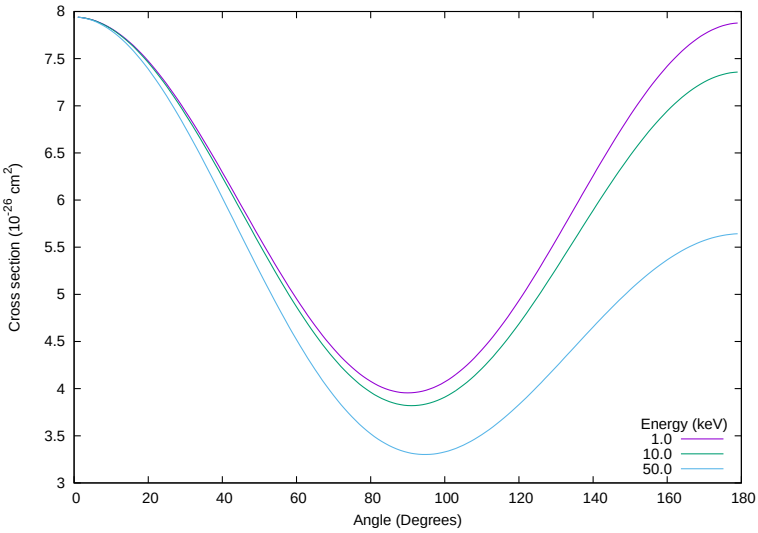


Figure 2.6: Klein-Nishina differential scattering cross section, shown for varying energy of incident photon.

The total Compton scattering cross section can thus be given by equation (2.18) (Rybicki and Lightman, 1985).

$$\sigma_s = 2\pi r_e^2 \left[\frac{1+x}{x^3} \left\{ \frac{2x(1+x)}{1+2x} - ln(1+2x) \right\} + \frac{1}{2x} ln(1+2x) - \frac{1+3x}{(1+2x)^2} \right], \quad (2.18)$$

where $x = \frac{h\nu}{m_e c^2}$

If the photon is determined to have scattered, it is given a new direction of propagation and it loses energy, the energy loss being calculated from equation (2.16). The likelihood of the photon scattering in any particular direction is given by the differential scattering cross section. However, each individual photon will need a specific scattering angle in order to continue its life in the simulation. This new direction of propagation relative to the photon propagation direction is found by rejection sampling the Klein-Nishina differential scattering cross section, equation (2.17). Rejection sampling is covered in more detail in Section 2.2.1.3. The rejection sampling returns the angle of scatter or in other words the polar angle in the photons coordinate frame. The photon's coordinate frame is a frame where the Z axis represents the photons direction of propagation. This is coupled with a new azimuthal angle, the azimuthal angle relative to the photon propagation direction is found by means of a uniform random deviate between 0 and 2π.

The photon's resulting direction of propagation is found by converting the direction of motion in the photon's coordinate frame into a direction of motion in the absolute coordinate system. This conversion is achieved through coordinate transformations. The details of the coordinate transformations are covered in Dunn and Shultis (2011). The important final result is shown in equation (2.19). If it is assumed that the direction vector (i, j, k) is the photon's direction vector before scattering, with (θ, ϕ) being the azimuthal and polar angles respectively, then $i = sin(\phi)cos(\theta)$, $j = sin(\phi)sin(\theta)$ and $k = cos(\phi)$. In order to get the direction vector after scattering, which is (i', j', k'), the transformation given by equation (2.19) must be applied. Where (θ_s, ϕ_s) are the azimuthal and polar angles of scatter respectively.

$$\begin{aligned} i' &= icos(\phi_s) + sin(\phi_s)\big(kcos(\theta)cos(\theta_s) - sin(\theta)sin(\theta_s)\big), \\ j' &= jcos(\phi_s) + sin(\phi_s)\big(ksin(\theta)cos(\theta_s) + cos(\theta)sin(\theta_s)\big), \\ k' &= kcos(\phi_s) - sin(\phi)sin(\phi_s)cos(\theta_s). \end{aligned} \tag{2.19}$$

2.2.4.2 Bound Scattering

As mentioned previously, it was originally thought that the high energy of the incident photons ($>$ 2 keV) meant the electrons could be treated as unbound. This is not the case. Molaro et al. (2014) showed that the binding effects cannot be ignored, specifically in the case of molecular hydrogen. As the clouds being simulated in the code are composed of molecular hydrogen rather than atomic, binding effects cannot be ignored. A comparison between bound and unbound scattering output spectra is shown in Chapter 3, figures 3.3 and 3.4.

As the electrons are not being treated as free, the code must also include Rayleigh scattering; in this context perhaps more accurately referred to as binding modified Thomson scattering. Rayleigh scattering is significant up to $\approx 20keV$ and like Thomson scattering, is elastic (the incident photon does not lose energy). Thus, for bound scattering the photons may either Rayleigh (elastic) or Compton (inelastic) scatter off hydrogen molecules and helium atoms in the cloud. As in the unbound scattering case, the scattering contribution of heavier elements is negligible and they are ignored.

The total scattering cross section becomes a combination of the Rayleigh and Compton bound scattering cross sections. The differential Rayleigh scattering cross section must be modified by what is known as the atomic form factor in

order to take into account the binding effects. Thus, it is taken to be the Thomson differential cross section by the atomic form factor, as given by equation (2.20) (Hubbell, 1997).

$$\frac{d\sigma_R}{d\Omega} = \frac{d\sigma_{Th}}{d\Omega} \times F^2(x, Z), \tag{2.20}$$

where $r_e = \frac{e^2}{m_e c^2}$ is the classical electron radius, $F^2(x, Z)$ is the atomic form factor, with Z being the atomic number of the absorber and $x = \frac{sin(\frac{\theta}{2})}{\lambda}$ is the momentum transfer variable with λ being the wavelength of the photon and θ the angle of scatter.

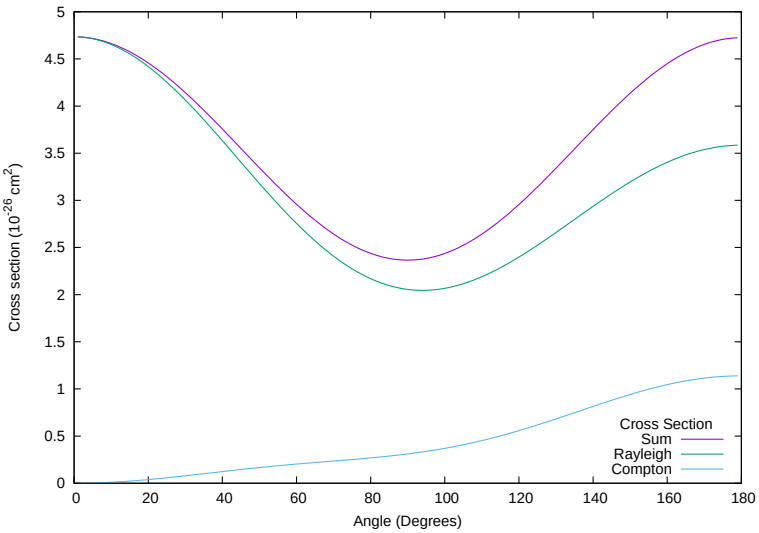
Atomic form factors are only possible to calculate analytically for hydrogen. However, Hubbell et al. (1975) calculated and tabulated the values for the heavier elements. The code makes use of these values through **xraylib** (Schoonjans et al., 2011), which is discussed in more detail in Section 2.2.5. Because the scattering being simulated is scattering from molecular hydrogen, not atomic, the Sunyaev et al. (1999) approximation is used and the Rayleigh scattering cross section is multiplied by a factor of two per electron.

The differential Compton scattering cross section is also modified and is taken to be the Klein-Nishina differential scattering cross section, modified by the incoherent scattering function, see equation (2.21) (Hubbell, 1997). These values are again from (Hubbell et al., 1975) and made use of in the code through **xraylib**.

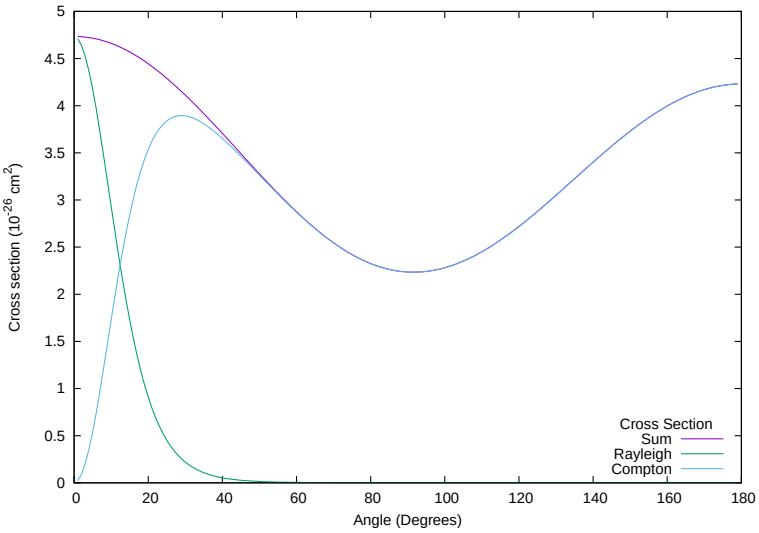
$$\frac{d\sigma_C}{d\Omega} = \frac{d\sigma_{KN}}{d\Omega} \times S(x, Z), \tag{2.21}$$

where $S(x, Z)$ is the incoherent scattering function. With Z being the atomic number of the absorber, $x = \frac{sin(\frac{\theta}{2})}{\lambda}$ the momentum transfer variable and λ the wavelength. $\frac{d\sigma_{KN}}{d\Omega}$ is given by equation (2.17). As for unbound scattering, if the photon Compton scatters it loses energy with the new energy being given by equation ((2.16)). The final bound differential scattering cross section is thus given by equation (2.22) and shown in Figure 2.7. Figure 2.7 shows how the individual contributions from Raylegih and Compton scattering combine, and how the contributions from each change as the energy of the incident photons changes. Rayleigh scattering rapidly begins to favour forward scattering as energy increases.

$$\frac{d\sigma_s}{d\Omega} = \frac{d\sigma_R}{d\Omega} + \frac{d\sigma_C}{d\Omega}, \tag{2.22}$$



(a)



(b)

Figure 2.7: Binding modified Rayleigh and Compton differential scattering cross sections for an incident photon of energy 1 keV (a) and 15 keV (b). Sum is the sum of both Cross sections.

2.2.5 Xraylib

Xraylib[1] is a freely available library of physical data concerning X-rays. Primarily intended for the study of X-ray fluorescence. it is written in ANSI C, can be linked to many other languages and compiled on many operating systems. It contains 40 callable functions that provide analytical and experimental data from many published works. The primary published source for **Xraylib** is Schoonjans et al. (2011). As mentioned in Section 2.2.4.2, **Xraylib** was used in this project as an easy way to access the data for the atomic form factors and incoherent scattering functions for binding modified scattering. The two functions that are used in the code are *CSb_Rayl()* and *CSb_Compt()*. These functions take energy and atomic number as arguments and return the value for the binding modified cross section in units of barn/atom. The computational overhead associated with the use of **Xraylib** was found to be quite minimal with total computational times increasing by no more than 5%.

2.2.6 Ray-Tracing and Geometry

Given the nature of multiple scattering a method was needed to calculate the distance to the edge of the cloud from any arbitrary photon position and direction of motion. For this the code makes use of a ray-tracing method, or perhaps more accurately, a ray-casting method. The code calculates all the points of intersection between the photon's direction of motion (line of movement) and the borders of the cloud (quadratic surface). Although any simple quadratic surface could be used with this method, only functions for spheres and flat capped cylinders have been written at this time. Simply put, the photon's direction of motion is taken to be a line in 3D, the equation of this line is then solved simultaneously with the equation of the shape. Solving the equations like this will lead to multiple points of intersection. The closest point of intersection in the photon's direction of motion is taken to be the point at which the photon will exit the cloud, or in the case of initial photon generation, the point at which the photon will enter the cloud.

The photon has a position (x, y, z) and a direction of motion (θ, ϕ). The parametric equation of the line is taken to be $p = o + vt$, where p is the photon's position, o is the origin of the line, v is the unit vector describing the photon's

[1] https://github.com/tschoonj/xraylib

direction of motion and t is an arbitrary scalar quantity of this vector. The equation of a sphere is $(x-c)^2 - r^2 = 0$, where x is a point on the sphere, c is the centre of the sphere and r is the radius of the sphere. These equations are solved simultaneously, such that $x = p$, substituting p into the equation of the sphere yields equation (2.23)

$$(o + vt - c)^2 = r^2, \tag{2.23}$$

which when expanded and rearranged gives equation

$$t^2(v \cdot v) + 2t(v \cdot (o-c)) + (o-c) \cdot (o-c) - r^2 = 0, \tag{2.24}$$

which is a quadratic of the form $at^2 + bt + c = 0$ in t, with constituents shown in equation (2.25),

$$\begin{aligned} a &= v^2 \\ b &= 2(v \cdot (o-c)) \\ c &= (o-c)^2 - r^2 \end{aligned} \tag{2.25}$$

Solving for t using equation (2.26).

$$t = \frac{-(v \cdot (o-c)) \pm \sqrt{(v \cdot (o-c))^2 - v^2(o-c)^2 - r^2)}}{v^2} \tag{2.26}$$

If there are no solutions for t then there is no point of intersection between the photon and the cloud. If there is one solution then there is one point of intersection, and if there are two solutions then there are two points of intersection. The important value to be found is the distance to the next point of intersection along the photon's direction of motion, which is taken to be the lowest positive value of t. This value is then taken as the scalar magnitude that gives the point of intersection and is used to calculate the distance between the photon's position and the point of intersection, which is then returned. The code of this function can be seen in Appendix B, the *spheredistance()* function.

The cylinder is solved in the same way. First the points of intersection between the photon's line and an infinite cylinder are found, then two circular caps are introduced to the ends of the cylinder. If points of intersection are found between these two circles the photons hit the cylinder. If any points of intersection intersect the circles, then the photons intersect the ends of the cylinder. The calculation is as follows. Again, the parametric equation of the line is taken to be

$p = o + vt$. The equation of an infinite cylinder is $x^2 + y^2 - r^2 = 0$. The ability to orientate the cylinder in any arbitrary direction is required, thus the equation of the cylinder is orientated on a line $p_c + v_c t = 0$. So the equation of the cylinder becomes equation (2.27) (Bielajew, 2000).

$$(q - p_c - (v_c \cdot (q - p_c))v_c)^2 - r^2 = 0, \tag{2.27}$$

where q is a point on the cylinder. Substituting q for the initial photon line $p + vt$ gives equation (2.28)

$$(p + vt - p_c - (v_c \cdot (p - p_c + vt))v_c)^2 - r^2 = 0, \tag{2.28}$$

which reduces to another quadratic equation in t with constituents given by equation (2.29) (Bielajew, 2000).

$$\begin{aligned} a &= (v - (v \cdot v_c)v_c)^2 \\ b &= 2((v - (v \cdot v_c)v_c) \cdot ((p - p_c) - ((p - p_c) \cdot v_c)v_c)) \\ c &= ((p - p_c) - ((p - p_c) \cdot v_c)v_c)^2 - r^2 \end{aligned} \tag{2.29}$$

Solving equation (2.29) for t would give the points of intersection with an infinite cylinder, however the cylinder needs end caps. Taking two circular caps with centres at m_1 and m_2, the centre of both caps will lie on the line that was used earlier to provide orientation to the cylinder. Thus, it can be taken that $m_1 = p_c$ and that $v_c = \frac{m_2 - m_1}{|m_2 - m_1|}$. From this it can be inferred that a finite cylinder is described by equation (2.27) and by the boundary conditions shown in equation (2.30)

$$\begin{aligned} (v_c \cdot (q - m_1)) &> 0 \\ (v_c \cdot (q - m_2)) &< 0, \end{aligned} \tag{2.30}$$

where q is the point of intersection with the infinite cylinder, found by solving equation (2.29).

The equation of a plane is $(p - m) \cdot v_c = 0$. Because the caps fit on the end of the cylinder, the vector normal to the plane is the same vector describing the orientation of the cylinder v_c, m is the centre of the cap and p is another point on the plane. Substituting the equation of the line for the incident photon and solving for t gives equation (2.31)

$$t = -\frac{((m - p) \cdot v_c)}{(v \cdot v_c)} \tag{2.31}$$

Solving for t for each cap will give the point of intersection with each plane. Finally, such a point q can be said to be on the cap when it satisfies the boundary conditions $(q - m_1)^2 < r^2$ for cap 1 and $(q - m_2)^2 < r^2$ for cap 2, where r is the radius of the cylinder/caps.

The code goes through the following procedure. First the two possible points of intersection with the cylinder are found by solving for t using the quadratic described in equation (2.29). If these points satisfy the boundary conditions in equation (2.30) then they are taken as points of intersection with the finite cylinder. The code then check for points of intersection on the end caps. This leaves a total of 4 possible points of intersection; the smallest non-negative of these points is taken as the photon's next point of intersection with the cloud boundary, the distance to which is the distance to the edge of the cloud.

2.2.7 Code Structure

The code contains two loops. A primary outer loop, which is the photon creation loop; every time it loops a new photon is generated. Within the primary loop is the secondary loop. The secondary loop handles the photon interactions, and will continue to loop through multiple scatter and absorption->re-emittance events, until the photon either escapes the cloud, is absorbed and not remitted, or the interaction limit is reached. The interaction limit is typically set to 35, so that is a maximum of 35 interactions for any one individual photon. The limit of 35 is a compromise between accuracy and computational time, in all but the most extreme of cases it will have no effect on code output. Again, code can be seen in full in Appendix B.

2.2.7.1 Non-Uniform Density

Unfortunately when modelling a non-uniform density, It is not computationally feasible to continually calculate the exact density at every single point in the cloud as the photon moves. As a compromise a method of 'steps' is used; a step is a small area of constant density. For example, in arbitrary units a distance of 10-15 from the cloud centre would have the same density, a distance of 15-20 would have a different density; the step size κ in this case being 5. The photon being propagated through these small areas of constant density. Decreasing the *stepsize* will increase the accuracy of the results, but will also increase computational time.

Figure 2.8 shows how this works in practice, showing both an actual Gaussian density distribution, and its stepped equivalent in the code.

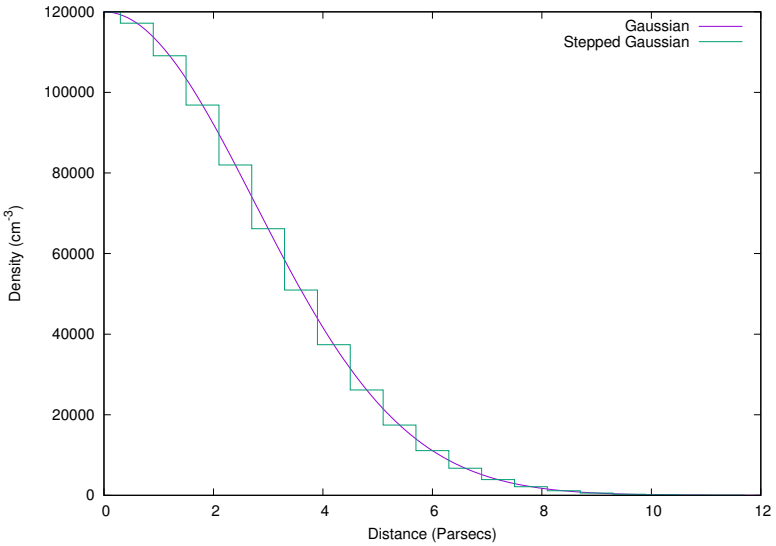


Figure 2.8: Actual Gaussian density distribution, and its equivalent stepped density distribution as used in the code.

2.2.8 Data Output

The code bins all of the photons that escape the cloud. The photons are binned both logarithmically by energy and by the direction angles (θ, ϕ) they have upon their escape. An important caveat is that photons that do not scatter at all, as in photons that pass directly though the cloud without interacting, are not binned. This makes the 0 degree case non-physical, as only photons that have scattered through very small angles will be binned in the 0 degree case. There is also a caveat concerning multiple scattering. Typically the maximum number of interactions within the cloud is set to 35 as mentioned in Section 2.2.7, photons that attempt to go over 35 interactions are not binned they are simply destroyed.

The energy bins are logarithmically spaced bins in the energy range E_{min} to E_{max}, which are typically set to 1 and 400 keV respectively. If a photon has energy pe, the bin i, into which it will be binned, is given by equation (2.32).

$$(int)i = n \times \frac{log(pe/E_{min})}{log(E_{max}/E_{min})}, \tag{2.32}$$

where n is the number of bins and i is rounded to the nearest integer. The number of bins is controlled by the global variable *ebins* and is typically set to 400.

For the angular binning, the photons are binned into areas of equal solid angle on a sphere. The surface area differential of a sphere is $dA = sin(\phi)d\phi d\theta$, thus the azimuthal angle θ is split into bins of equal size between 0 and 2π, while the cosine of the polar angle $cos(\phi)$ is split into bins of equal size between 0 and π. Typically the number of angular bins is set to 180, making the solid angle of each bin 5.317×10^{-6} sr. 180 angular bins was chosen so as to ensure reasonable counts in all bins, while still having the necessary resolution. If the equivalent width of the iron line were a focus of this project the number of angular bins would have been increased, certainly around the iron line itself.

When outputting the data, each energy bin takes the energy of the centre of that bin, which is found by means of equation (2.33).

$$binenergy = E_{min} \times \left(\frac{E_{max}}{E_{min}}\right)^{i/n}, \tag{2.33}$$

where i is the bin in question and n is the number of bins.

The data output is written in tab separated columns, which from left to right are energy, azimuthal bin, polar bin. Turning this column data into a spectrum is achieved by means of a **python** script which is shown and described in Appendix B.2

2.3 Code Testing

Testing was performed for various parts of the code. In the interest of clarity in the operation of some of the parts of the code some test results are included in this section.

Figures 2.9 to 2.11 are 3D plots that show how the ray-casting from Section 2.2.6 works in practice. Figure 2.9 shows a sphere illuminated by a source at its centre, where each point is a location calculated by the ray casting function. Figure 2.10 shows an externally illuminated sphere. Bear in mind the different scales on the y and z axis compared to the x. Finally, Figure 2.11 shows an internally illuminated cylinder

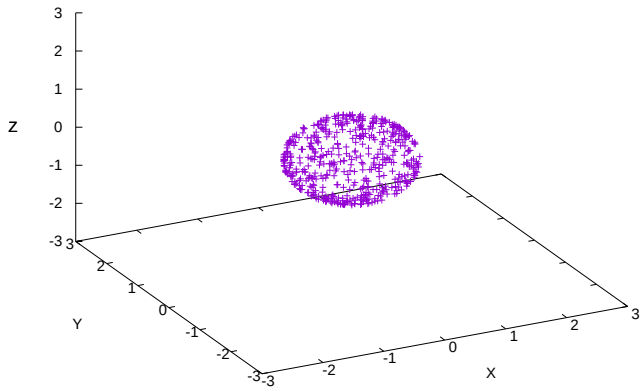


Figure 2.9: Ray cast testing, sphere illuminated internally.

For testing the rejection sampling, the number of iterations will play a large part in how well the distribution appears to be re-created. Figure 2.12 shows the Klein Nishina differential scattering cross section re-created at an energy of 5 keV. Figure 2.13 shows it for an energy of 50 keV. The noise in the rejection sample is reduced by increasing the iterations, Figure 2.14 is shown with the iterations increased by a factor of 5.

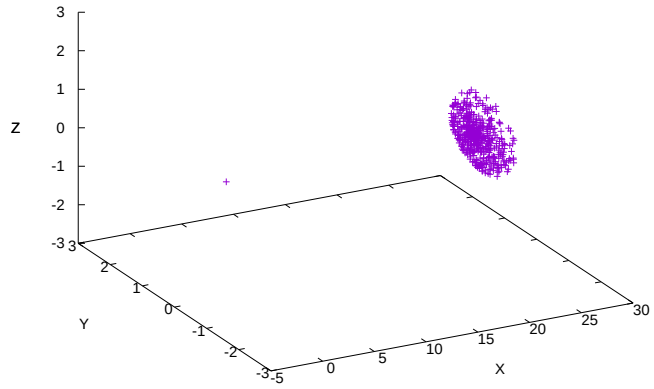


Figure 2.10: Ray cast testing, sphere illuminated externally, the source is located at (0, 0, 0).

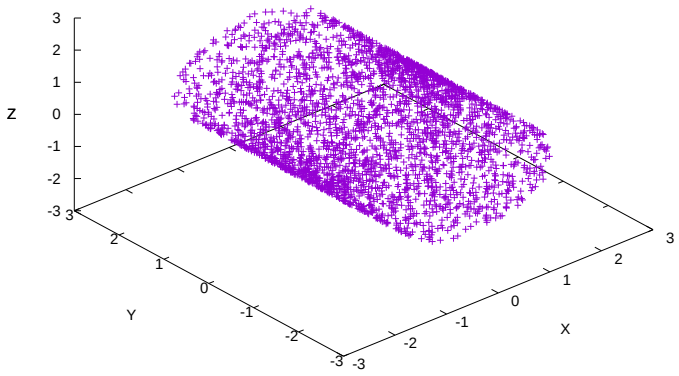


Figure 2.11: Ray cast testing, cylinder illuminated internally.

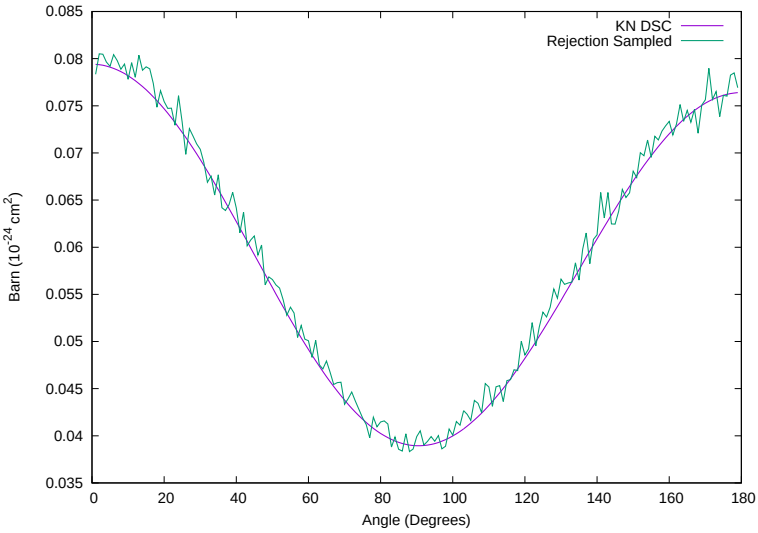


Figure 2.12: Rejection sampled Klein-Nishina differential scattering cross section and actual values. Energy = 5 keV.

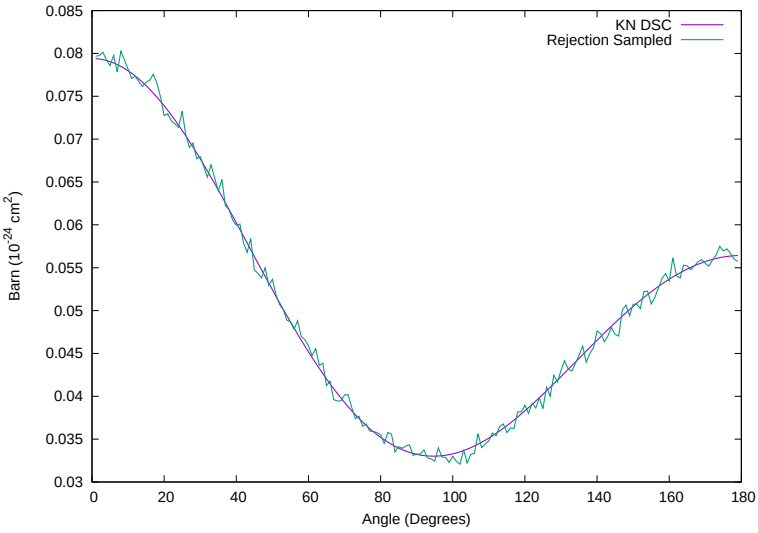


Figure 2.13: Rejection sampled Klein-Nishina differential scattering cross section and actual values. Energy = 50 keV.

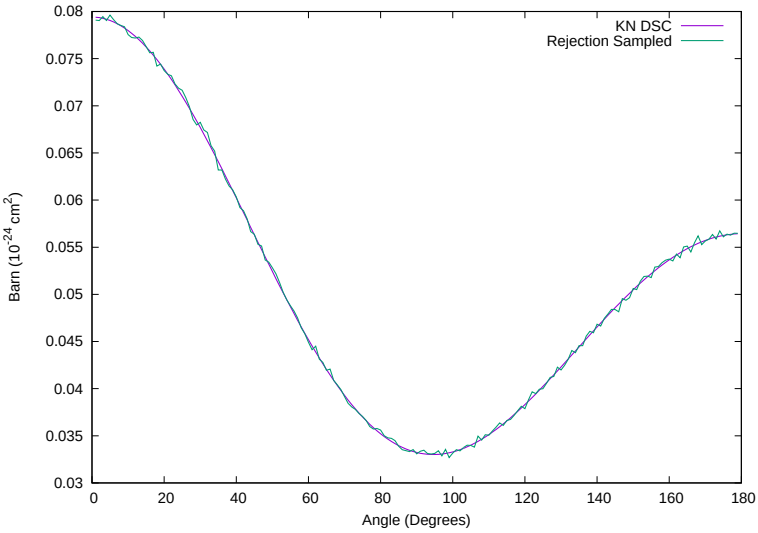


Figure 2.14: Rejection sampled Klein-Nishina differential scattering cross section and actual values. Energy = 50 keV. Samples increased by factor of 5.

Chapter 3

Code Output Analysis

In this chapter the output of the Monte Carlo code described in Chapter 2, will be analysed and explained in detail. Beginning with a comparison of the code output with analytical calculations, followed by an analysis of the continuum shape and the iron line and finally, concluding with a discussion on time dependence effects.

3.1 Analytical Comparisons

As mentioned in Chapter 1 and Section 2.2.4, the X-ray reflection spectra of a Compton thin, uniform density sphere can be calculated analytically (see the appendix of Walls et al. (2016) for details). These analytically calculated spectra can be used to verify the correctness of the output of the Monte Carlo code.

When considering only Klein-Nishina scattering (Section 2.2.4.1) and putting the code into single scatter mode, the Monte Carlo code is in good agreement with the analytical calculation. Figure 3.1 shows a comparison between the analytical calculation and the output of the Monte Carlo code, in both single and multiple scatter mode, for a Compton thin spherical cloud ($N_H = 5 \times 10^{22}$ cm^{-2}). The analytical calculation and the single scatter code output are in agreement. There is a slight discrepancy introduced when allowing for multiple scattering. However, as the cloud has a low N_H there is not much multiple scattering occurring, meaning the discrepancy is quite small (a factor of about 1.2).

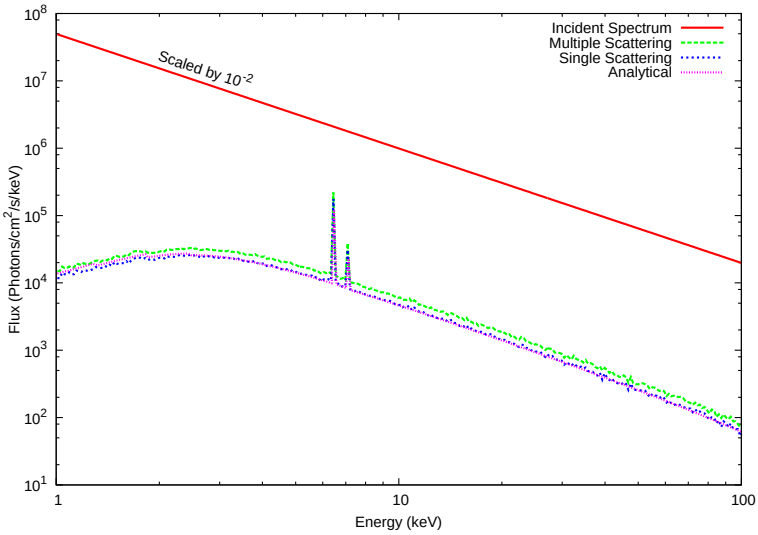


Figure 3.1: Comparison of the spectra produced by the Monte Carlo code when considering only Klein-Nishina scattering in single and multiple scatter modes, versus an analytically calculated spectrum for a 2 pc diameter uniform density sphere. Line of sight angle $\theta = 120°$. $N_H = 5 \times 10^{22}$ cm^{-2}. The incident spectrum has been downscaled by 10^{-2} to aid visualization.

The discrepancy becomes far more pronounced as N_H increases. Figure 3.2 shows the same comparison as Figure 3.1, but for a Compton thick cloud ($N_H = 5 \times 10^{24}$ cm^{-2}). With the increased likelihood for multiple scattering to occur, the spectral shape and flux will change significantly between the analytical calculation and the Monte Carlo code output. This underlines the importance of using the Monte Carlo technique. Furthermore, from this comparison it can be confirmed that the Monte Carlo code is working as intended. Additional confirmation can be seen by comparing to similar works, for example Odaka et al. (2011) Fig 7.

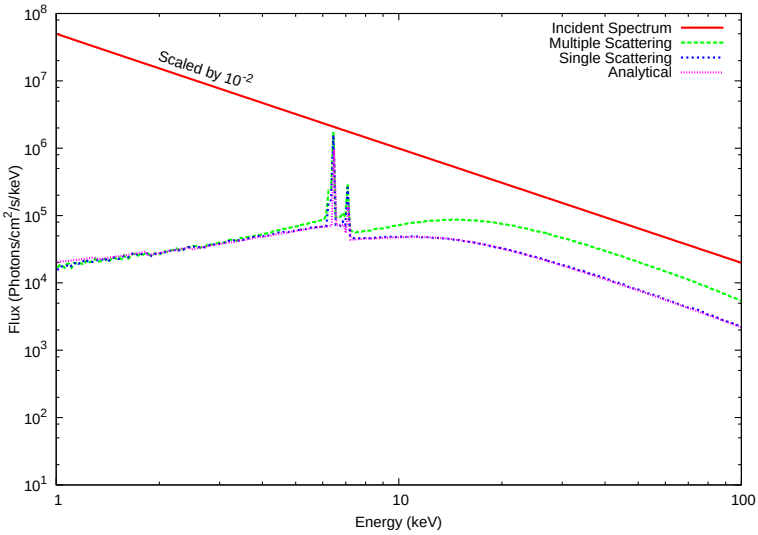


Figure 3.2: Comparison of the spectra produced by the Monte Carlo code when considering only Klein-Nishina scattering in single and multiple scatter modes, versus an analytically calculated spectrum for a 2 pc diameter uniform density sphere. Line of sight angle $\theta = 120°$. $N_H = 5 \times 10^{24}$ cm^{-2}. The incident spectrum has been downscaled by 10^{-2} to aid visualization.

3.2 Continuum

As discussed in Section 2.2.4.2, treating the electrons as free is not possible, bindings effects are quite important in the low energy regime. Figure 3.3 shows a comparison of the total scattering cross section for free electrons versus that for bound electrons. It shows that for atomic hydrogen the actual difference between the two is minimal (approximately 2%). However, when the fact that the photons are actually scattering from molecular hydrogen is taken into account, the result changes quite drastically. This change is a result of a modification to the Rayleigh scattering cross section. It has been increased by a factor of two as per Sunyaev et al. (1999), who give a factor of two increase as a good approximation for the Rayleigh scattering cross section for molecular hydrogen. This manifests itself as a large increase in scattering below 20 keV and leads to a divergence in continuum shape for the output spectra, which can be seen in Figure 3.4.

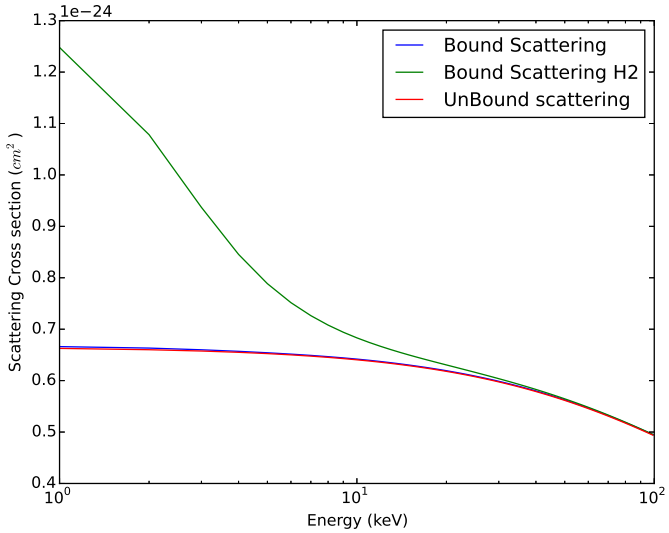


Figure 3.3: Comparison of the total hydrogen scattering cross section for unbound electrons, bound electrons from atomic hydrogen and bound electrons from molecular hydrogen.

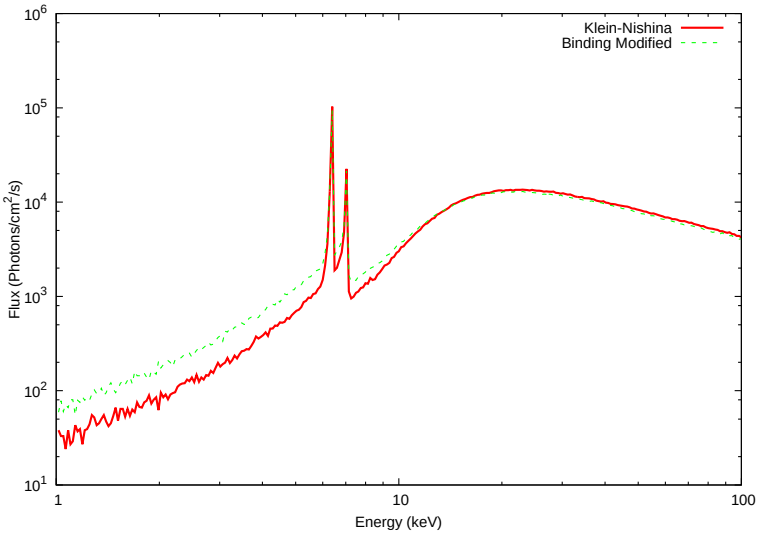


Figure 3.4: Comparison of the spectra produced by the Monte Carlo code using Klein-Nishina scattering, versus binding modified Rayleigh and Compton scattering for a 2 pc diameter uniform density sphere. Line of sight angle $\theta = 30°$ and $N_H = 5 \times 10^{24}$ cm^{-2}.

3.2.1 Angle

The spectral shape and total flux of the scattered emission is very dependent on the relative position of the cloud. Figure 3.5 shows the changes in the output spectrum induced by a changing line of sight angle θ. A number of observations can be made from Figure 3.5. The 0 degree case has hugely increased low energy absorption, due to the fact that observed photons must traverse the entire cloud before escaping. This would be similar to a spherical cloud that is surrounding the continuum source (Leahy and Creighton, 1993).

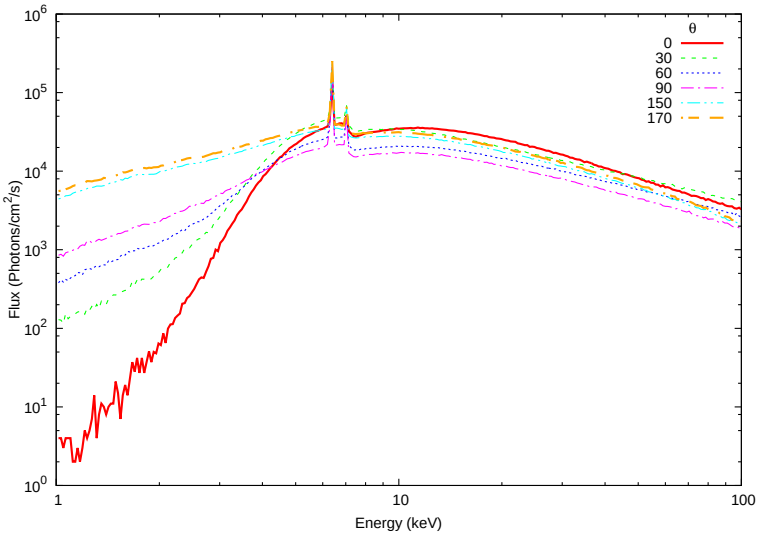


Figure 3.5: Spectra produced by the MC code for a 2 pc diameter, uniform density sphere with an N_H of 6×10^{23} cm^{-2}, and a photon index of 2.0. Showing the changes in flux and continuum shape resulting from a changing line of sight angle.

Iron edge depth decreases with increasing angle, due to absorption being less prominent at higher angles compared to scattering. As with the large increase in total continuum flux with increasing angle, it is caused by an interplay of two separate effects. The first being, at energies $1 - 50$ keV, the Compton scattering distribution is Thomson-like, i.e. photons are as likely to scatter backward as forwards. The second is due to the fact that most photons are scattered from the cloud from a very shallow depth. After a photon is scattered in the direction of the observer, it is progressively less likely to be absorbed before escaping as the scattering angle increases. It can be seen in Figure 3.5 that, as energy increases, the disparity in flux reduces and reverses. Note the reversal in flux between the 60 and 150 degree cases around 60 keV, which is where the Compton scatter-

ing distribution becomes non-Thomson in nature and begins to favour forward scattering.

The large increase in flux for the 0 degree case is not caused by the same process, due to the fact that the energy is too low for forward scattering to be favoured. However, at energies > 10 keV absorption becomes increasingly unlikely. Thus photons are scattering not just on the surface of the cloud, but have the chance to pass through the entire length of the cloud, leading to increased flux at energies > 10 keV. It should be noted that, as discussed in Chapter 2, photons that do not scatter at all do not contribute to the spectrum. The primary effect this has on the 0° spectrum is a massive reduction in observed flux. Furthermore, in the case of a low density cloud, the incident power law itself would be observed in the 0° case.

3.2.2 Column Density

Considering the case of varying column density N_H which can be seen in Figures 3.6 & 3.7; In the high angle case (Figure 3.6), it can be seen that low energy absorption is unaffected by increasing N_H due to the fact that all the photons are back scattered from the cloud from a very shallow depth. The effect is far less pronounced in the low angle case (Figure 3.7). In the high angle case, at low energies (< 10 keV), differences in continuum shape and flux become very small after a critical N_H is reached ($N_H > 5 \times 10^{23}$ cm^{-2}). This could potentially lead to issues in N_H determination if the line of sight angle is quite high.

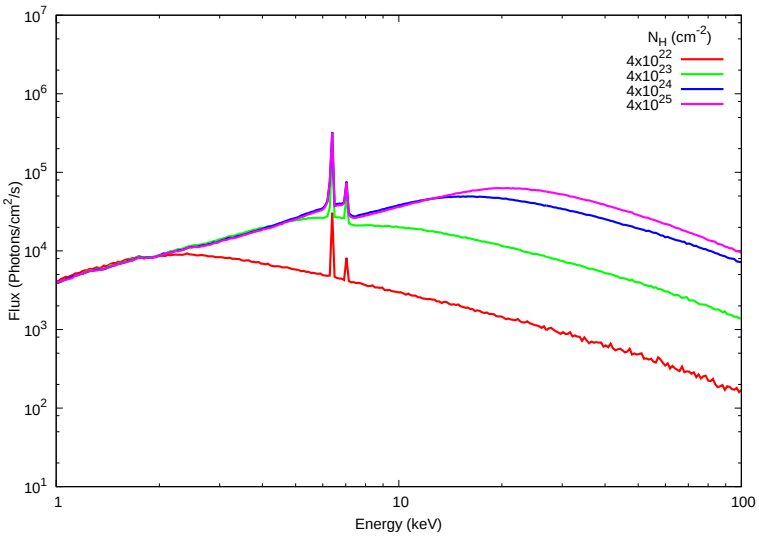


Figure 3.6: Spectra produced by the MC code for a 2 pc diameter, uniform density sphere positioned with a line of sight angle $\theta = 140°$, and a photon index of $\Gamma = 2.0$. Showing the changes in flux and continuum shape resulting from a changing N_H.

Of particular note is the column density induced flux differences. In the low angle case (Figure 3.7) the high column density $N_H = 5 \times 10^{25}$ cm^{-2} has very low flux in the high energy (> 10 keV) compared to the lower column density cases. This counter-intuitive result is found to be due to increased multiple scattering, where the extra scattering leads to scattering away from the observer, with the effect becoming more pronounced as multiple scattering increases. This effect is shown clearly in Figure 3.8, where absorption has been removed and only scattering is in effect. The first plot shows the output where the photons are only allowed to scatter once, demonstrating that as N_H increases, a saturation point is quickly reached where every photon that can scatter does scatter. The

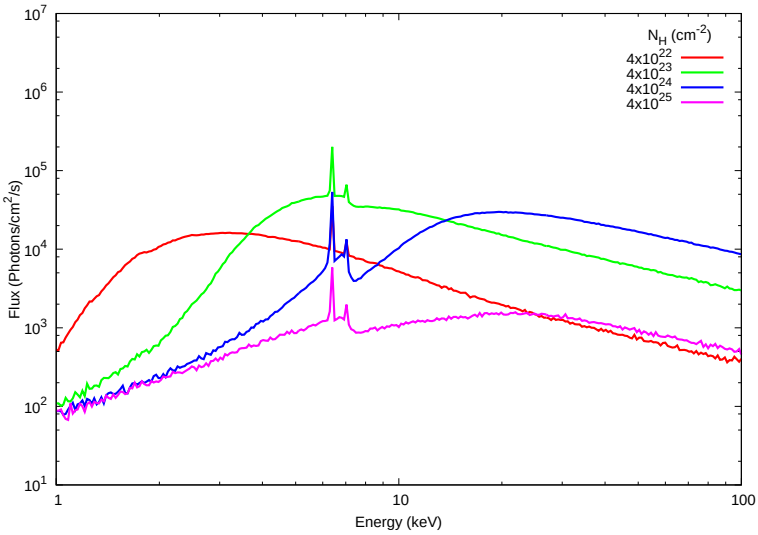
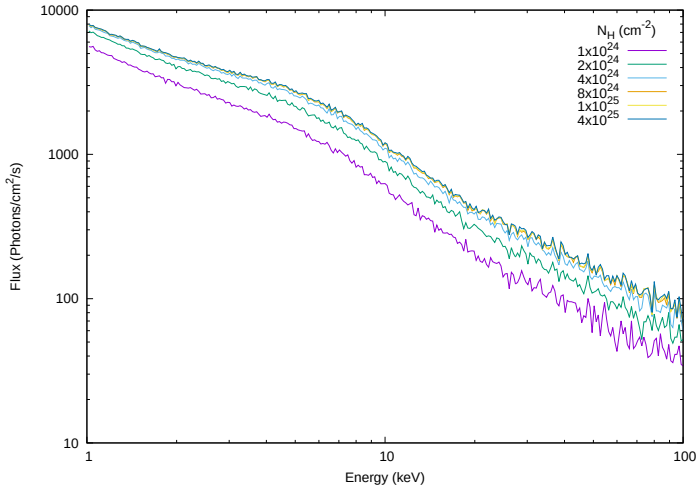
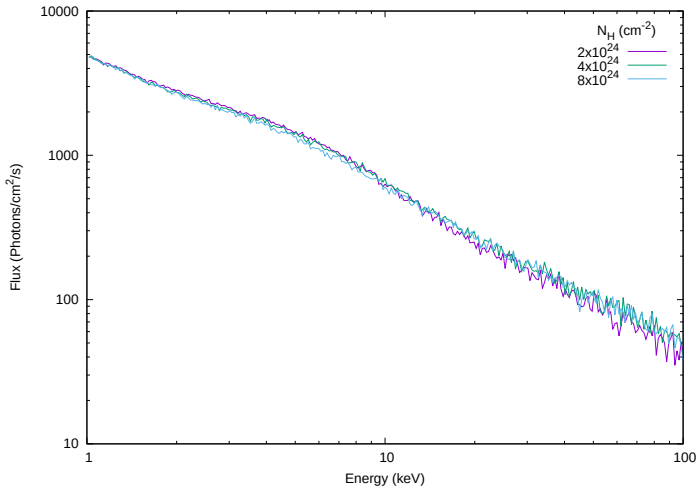


Figure 3.7: Spectra produced by the MC code for a 2 pc diameter, uniform density sphere positioned with a line of sight angle $\theta = 30°$, and a photon index of $\Gamma = 2.0$. Showing the changes in flux and continuum shape resulting from a changing N_H.

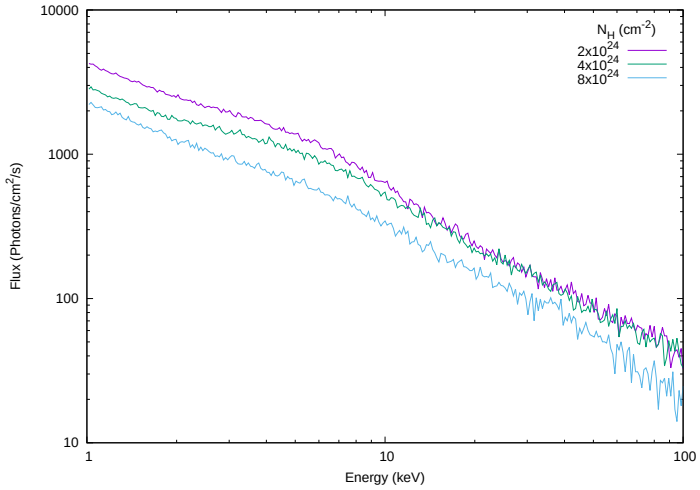
second plot shows the output where the photons are allowed to scatter twice, demonstrating that multiple scattering not only leads to a decrease in total flux, but that the lower N_H spectra rise in relative flux compared to the higher N_H spectra. However, as multiple scattering is increased to 10 allowed scatters for each photon (the third plot), it is clear that the higher N_H spectra begin to have lower relative flux. This is in agreement with the actual code output as seen in Figure 3.7.



(a)



(b)



(c)

Figure 3.8: Flux and relative flux changes in continuum, for a changing column density. Absorption has been disabled. The first to last figures representing a) single scattering, b) two scatters and c) 10 scatters respectively. $\theta = 30°$.

3.2.3 Photon Index

In the case of changing photon index Γ, the changes in low energy continuum shape are not so significant. However, the slope of the high energy (> 10 keV) does change quite a lot with changing photon index. This can be seen in Figure 3.9, where the index is varying from 1.4 to 2.8. The implication being that low energy data (< 10 keV) will be unlikely to determine the photon index with great accuracy, but that higher energy data may be able to.

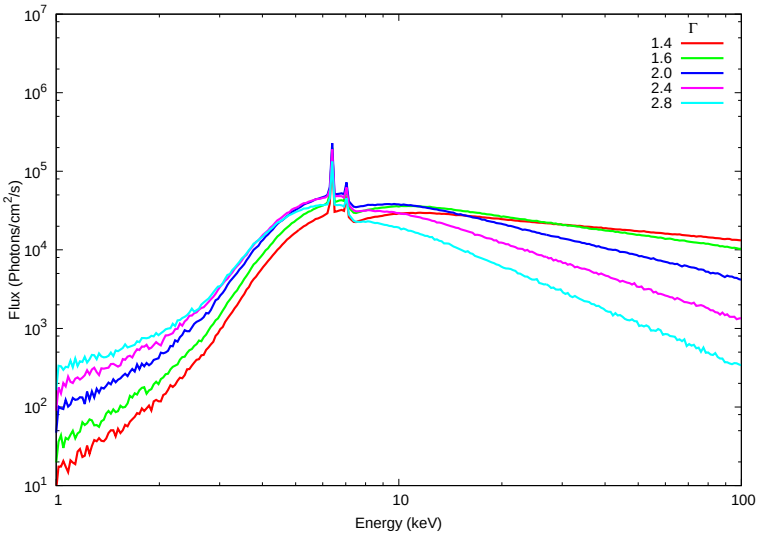


Figure 3.9: Spectra produced by the Monte Carlo code for a 2 pc diameter, uniform density sphere positioned with a line of sight angle $\theta = 30°$ and $N_H = 6 \times 10^{23}\ cm^{-2}$. Showing the changes in flux and continuum shape resulting from a changing photon index Γ.

3.3 Iron Line Flux

Figure 3.5 would suggest that after an angle of about 90°, the shape of the continuum is unlikely to be a particularly viable way of determining the angular position of the cloud. However, the 6.4 keV K_α iron line can be used as well. The relative strength of the iron line flux versus the line of sight angle is shown in Figure 3.10, peaking at $\theta = 90°$; with the relative flux being the strength of the iron line above the continuum. This shows a clear angular dependence for the iron line. Importantly, the angular dependence of the iron line strength is independent of the column density of the reflecting cloud.

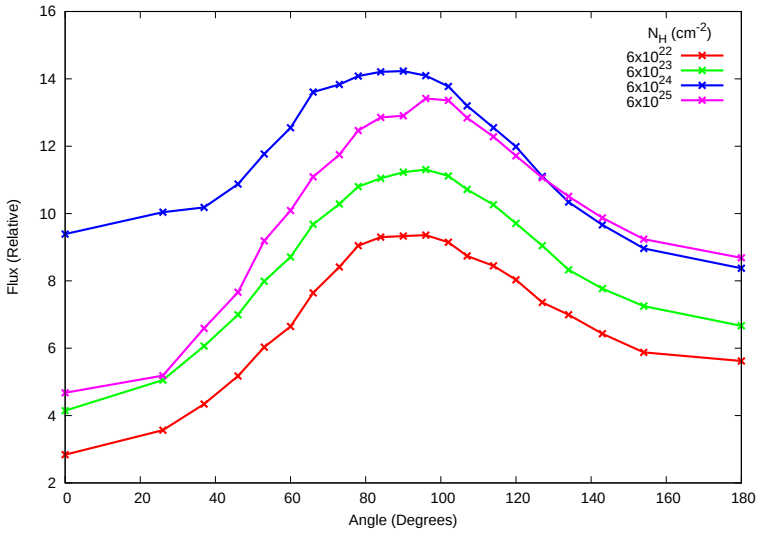


Figure 3.10: The dependence of the relative strength of the Fe K_α line for different N_H values versus varying angle θ.

The iron line can also be helpful in determining the column density N_H. Figure 3.11 shows how the total iron line flux varies with increasing column density N_H. There is an increase in flux up to an N_H of 6×10^{23} cm^{-2} and then a gradual decrease. This confirms results from Sunyaev and Churazov (1998) who gave a maximum Thomson optical depth of ≈0.4 for maximum iron line flux. Interestingly, there is a plateauing of the flux as angle increases, which is due to the fact that only the surface of the cloud is being 'seen' at higher angles.

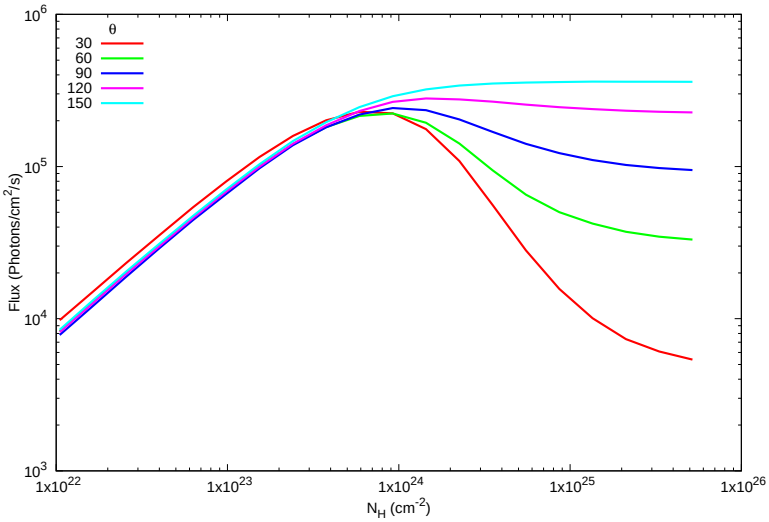


Figure 3.11: The dependence of the total Fe K_α line flux versus varying N_H for various angles.

3.4 Cylindrical Cloud

As mentioned in Chapter 2, the code can also model a cylindrical cloud. Although a cylindrical cloud will not be made use of further, the results of its output are presented in this section for the sake of completeness. Figure 3.12 shows a comparison between a spherical cloud and a cylindrical cloud of equal mass. Initially, Figure 3.12 would suggest that X-ray data would be easily able to differentiate between a cloud that is spherical and one that is cylindrical, as the discrepancy between the two is quite large. However, there are a few things to consider. Primarily, the difference between the two spectra is not all dissimilar to the difference caused by a changing angle. Thus, it is unlikely that a fitting procedure, as will be performed in this project, would be able to distinguish between the two. However, there is also the orientation of the cylinder to consider. Figure 3.13 shows a comparison for the same cylindrical cloud in different orientations. In the lower part of Figure 3.13 the illuminating source is inside the cloud; this is for illustration purposes only. The orientation does not have a large effect on spectral output, except for in the face on case (the 0 degree case), where photons must traverse a far larger N_H (the length of the cylinder), leading to massively increased absorption. It is for these two reasons it would likely be necessary to discern the geometry of the cloud being fit beforehand, in order to reduce the number of parameters in the fit.

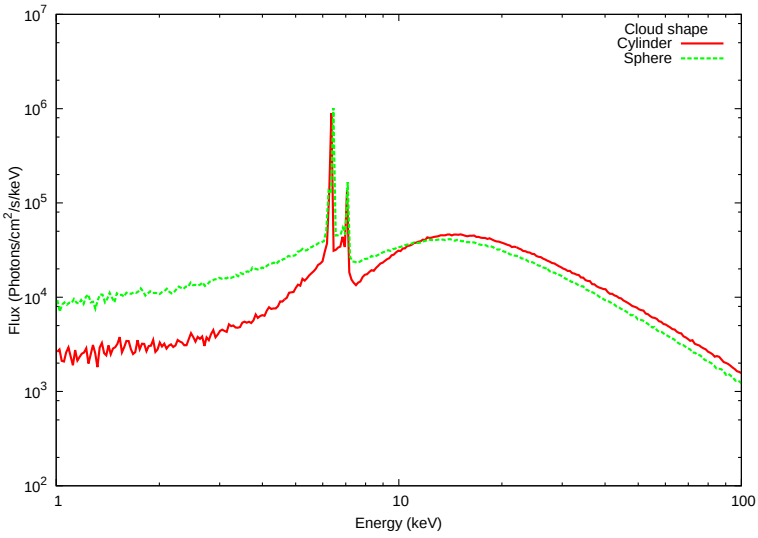


Figure 3.12: Comparison of a spherical and a cylindrical cloud of equal mass.

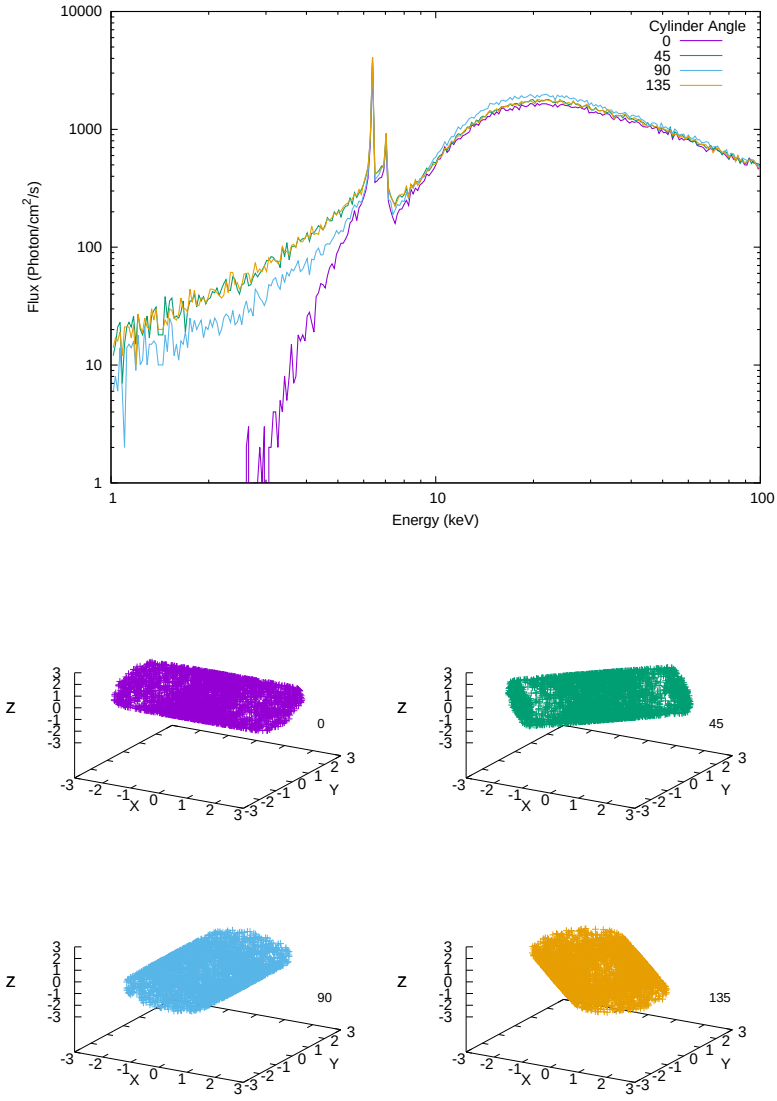


Figure 3.13: Cylinder reflection spectra for cylinders with changing orientation. The cylinder is at a line of sight angle of 30°. The orientation angle is shown in the legend, and beside each cylinder.

3.5 Non-Uniform Density

As discussed in Chapter 2, the code is able to model density profiles other than uniform, currently being able to calculate spectra for r^{-2}, e^{-r} and Gaussian density profiles. These profiles have been chosen to provide a broad spectrum of different density profiles. Figure 3.14 shows a comparison between density profiles for spherical clouds, with the line of sight angle held constant. The Gaussian profile has a standard deviation of $\sigma = 1.1$ pc and a radius of 4 pc. The clouds are of constant mass, with central density ρ_0 being equal across profiles and the radius of the cloud changing. This is a better comparison for different density profiles than spheres of constant radius, as the N_H would be quite different for each of them. The smaller plot in the top left of Figure 3.14 is the iron line, however it has been normalised by incident flux. Incident flux is changing because the number of generated photons remains constant, but the size of the cloud does not.

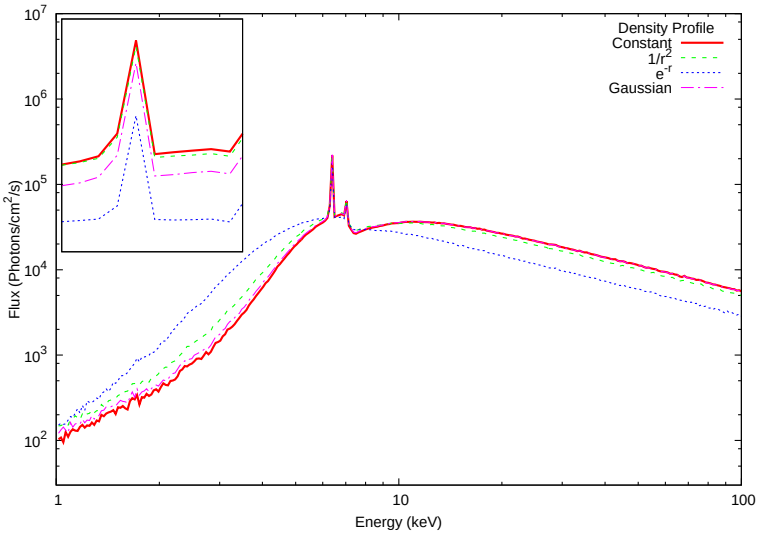


Figure 3.14: Comparison of the four utilized density profiles for a sphere of constant mass, radius is changing, the line of sight angle $\theta = 30°$ and $\rho_0 = 1.2 \times 10^5$ cm^{-3} ($N_H = 2.5 \times 10^{24}$ cm^{-2}). The variable density cases have a step size $\kappa = 0.1$ pc. The smaller figure in the top left showing the iron line, has been normalised by incident flux.

Of note is how the reflected spectrum is not changing drastically between density distributions, only the e^{-r} distribution shows a significant discrepancy. When normalizing for incident flux, as shown in Figure 3.14, relative flux changes

increase. It is these different normalisation values that will lead to different fitting results when dealing with non-uniform density clouds. As shown in Chapter 5.

Figure 3.15 demonstrates that changes in the line of sight angle are still readily apparent for the Gaussian case. Notably, the point at which the 30 degree and 90 degree continuum flux reverse is a few keV lower than for the case of the uniform density. This is due to surface scattering no longer being as dominant, with the photons being able to penetrate further into the cloud on average than the uniform density case.

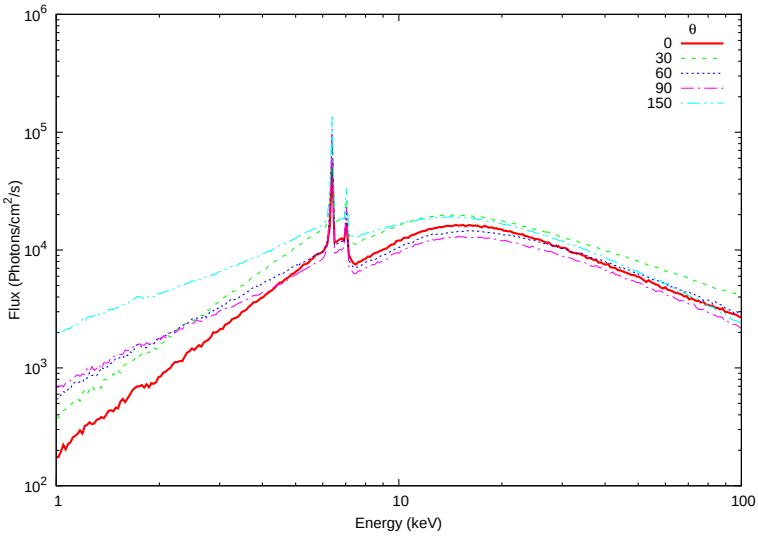


Figure 3.15: Similar comparison as that shown in Figure 3.5, except for a variable density sphere. A Gaussian density profile, with a central density of $n_0 = 1.2 \times 10^5$ cm^{-3} ($N_H = 2.5 \times 10^{24}$ cm^{-2}), $\sigma = 1.1$ pc, and a step size of 0.1 pc. Showing the changes induced by a changing line of sight angle.

3.5.1 Step Size

As mentioned in Section 2.2.7.1 the non-uniform density code makes use of a 'step size', which is steps of constant density, in order to approximate the non-uniform density of the cloud. The smaller the step size the more accurate the simulation, but the increase in computational time associated with decreasing step sizes can be excessive. Figure 3.16 shows a comparison of output spectra for a cloud with a Gaussian density distribution. The line of sight angle is 30 degrees, the central density is $\rho_0 = 1.2 \times 10^5$ cm^{-3} and the radius is 4 pc. The step sizes are shown as a percentage of the radius of the cloud. It is clear that once the step size is below approximately 5% of the radius of the cloud there is no need to lower the

step size further. The case of the step size being 30% of the radius of the cloud for example, has a low energy discrepancy of a factor of ≈ 50. The reduces as energy increases and a saturation point of scattering is reached.

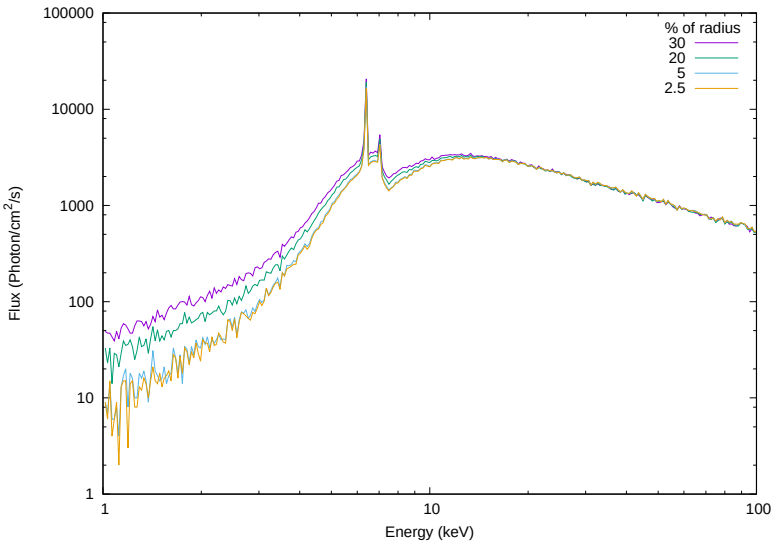


Figure 3.16: Comparison of output spectra for a Gaussian density distribution cloud, showing the effect of different step sizes. Step sizes are shown as a percentage of the radius of the cloud.

However, this is only for the simple case of a spherical cloud, and a very well defined density distributions. Although no analysis is done in this project, it is possible that more complex cloud shapes and/or density distributions would require steps sizes to be smaller. In this case a more variable step size would likely be required, perhaps based on the interaction length.

3.6 Iron Abundance

As mentioned in Section 2.2.3 the code can quite easily vary the relative metallicity of the cloud, in particular the iron abundance. Figure 3.17 shows the effect of increasing iron abundance on the spectral output. Figure 3.18 shows the effect of increasing iron abundance on the relative strength of the iron line for varying angular positions. Both figures are for a Compton thick cloud.

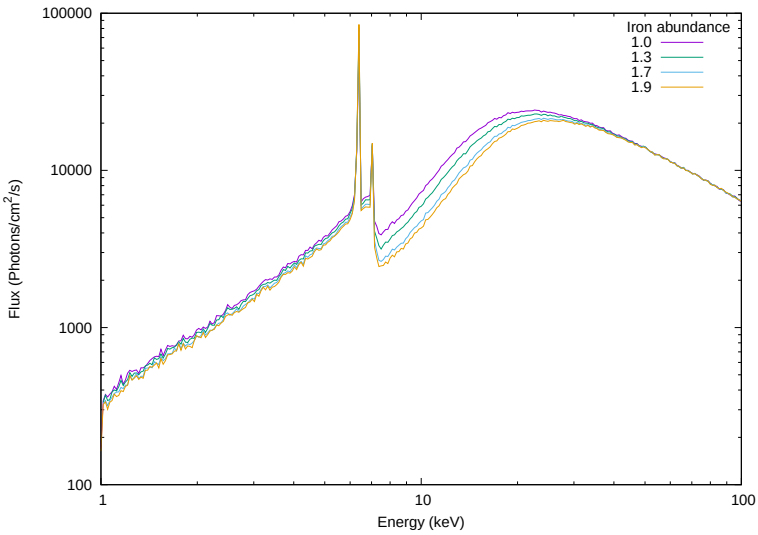


Figure 3.17: Comparison of output spectra for a Compton thick cloud at a line of sight angle of 60°, showing the changes induced by an increasing iron abundance. Iron abundance is shown as a factor of Solar abundance (see Table 2.1).

The iron abundance does not significantly affect the low energy continuum ($<$ 10 keV). There is a small decrease in total scattered flux in the low energy due to increased absorption. In fact, it does not have a significant effect on the continuum until an energy of approximately 7 keV, at the iron K-edge, with increasing abundance obviously leading to large increases in the size of the K-edge. These differences continue up to approximately 40 keV where all differences then disappear. Good quality data would be needed to analyse the K-edge and determine the iron abundance. Figure 3.18 shows how the relative strength of the iron line will increase as iron abundance increases, showing a linear increase in line strength; this could be analysed more easily. The differences are, however, still relatively minor, being far smaller than the differences induced by changing angle, for example (Figure 3.10).

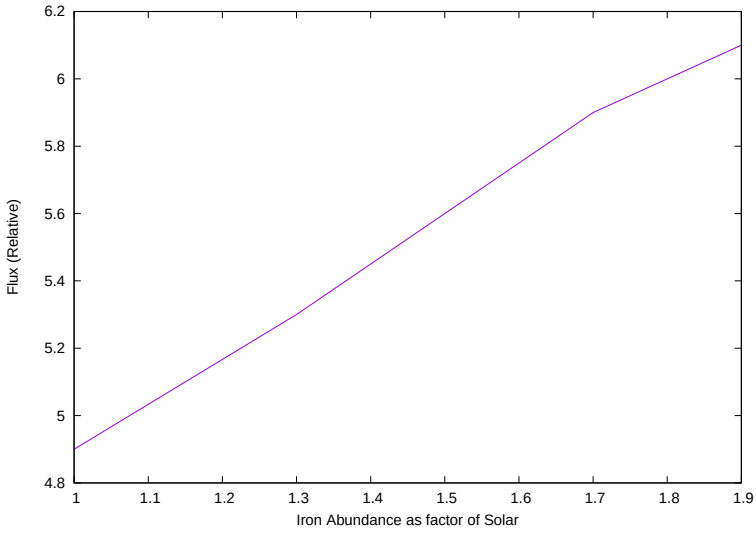


Figure 3.18: Dependence of the relative iron line strength above the continuum on an increasing iron abundance, for a Compton thick cloud at a line of sight angle of 60°. Iron abundance is shown as a factor of Solar abundance (see Table 2.1).

3.7 Time Dependent Effects

The question of time dependence is a complex one. The code makes two approximations related to time dependence. Firstly, that there is no duration to the initial photon release, that is to say that all the photons are released at the same instance. The code can be thought of as providing a snapshot of an incident flare's reflected component. The second is that all photons that escape the cloud will be included in the final output spectra regardless of time spent in the cloud. This is a non-physical situation as all of the photons escaping the cloud will in reality be escaping at vastly different time intervals. In this section, several effects of time dependence will be analysed and discussed, with their affect on the final results being discussed in Chapter 6.

3.7.1 Escape Time

The escape time from the cloud of any individual photon is dependent on two primary factors, namely, the size of the cloud and the number of scatters/re-emissions. Secondary factors include the geometry of the cloud, the entry point of the photon, and in terms of observed photons, the position of the cloud. The average escape time of any particular photon will decrease as the density increases.

Figure 3.19 shows this effect with the average escape time decreasing rapidly with increasing density. This is for two reasons: the increased likelihood to scatter with increasing density and the increased absorption 'destroying' more photons.

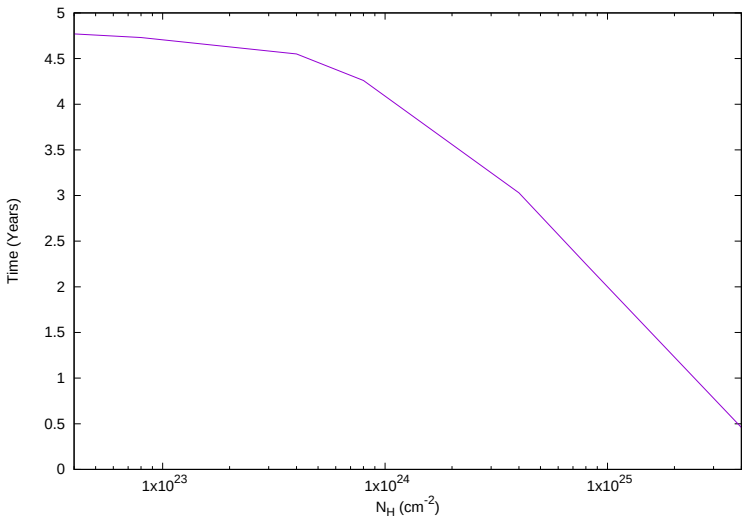


Figure 3.19: How average escape time from a 2 pc diameter cloud increases with increasing cloud cloud column density N_H.

However, there is a smaller number of photons whose escape time will dramatically increase as density increases. These are photons that are multiply scattering within the cloud. Figure 3.20 shows the percentage of escaped photons versus the number of successive scatters for a Compton thick cloud. Over 50% of all photons will escape the cloud after scattering only once, with 80% of all photons escaping after their 3rd successive scatter.

Figure 3.21 shows the average escape time of photons versus the number of successive scatters the photons have undergone. Included is data for three N_H values. Reinforcing what Figure 3.19 shows, increasing the N_H leads to a large decrease in average escape time. Note that the low N_H case only goes up to 6 scatters due to low multiple scattering at this density level. As the number of scatters increases, so does the average escape time, regardless of the N_H. There is a large difference between the $N_H = 4 \times 10^{24}$ cm^{-2} and $N_H = 4 \times 10^{25}$ cm^{-2} resulting from the fact that at $N_H = 4 \times 10^{25}$ cm^{-2} the cloud is so dense, nearly all of the multiple scattering is happening right at the surface of the cloud.

The primary effect that this increasing escape time with multiple scattering has, is to delay the release of photons from the cloud. This could potentially lead to issues with the validity of the code (The effect is discussed in detail in

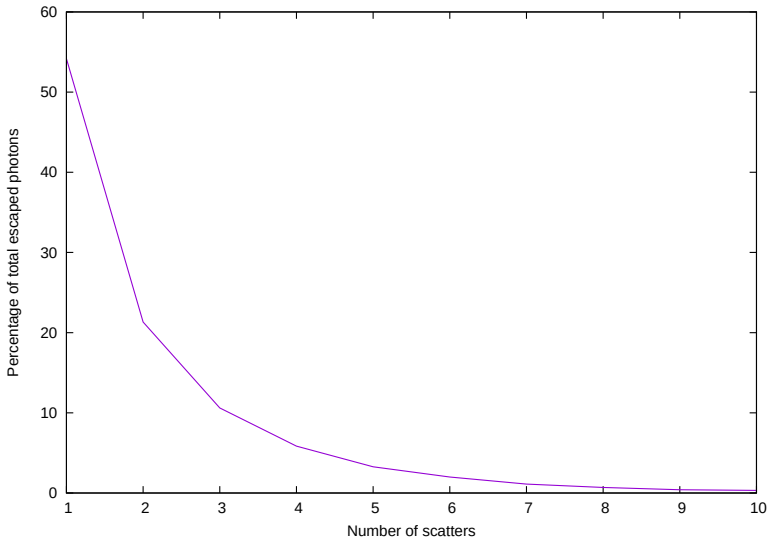


Figure 3.20: Number of successive scatters as a percentage of total escaped photons from a Compton thick cloud.

Sunyaev and Churazov (1998)). As mentioned previously, the code is treating the observed photons as if they have all left the cloud at the same time. Evidently, this is a non-physical situation, a not insignificant number of the photons that contribute to the output spectrum should actually be released at far later time periods due to them having multiply scattered. The impact on spectral shape is discussed in the following section.

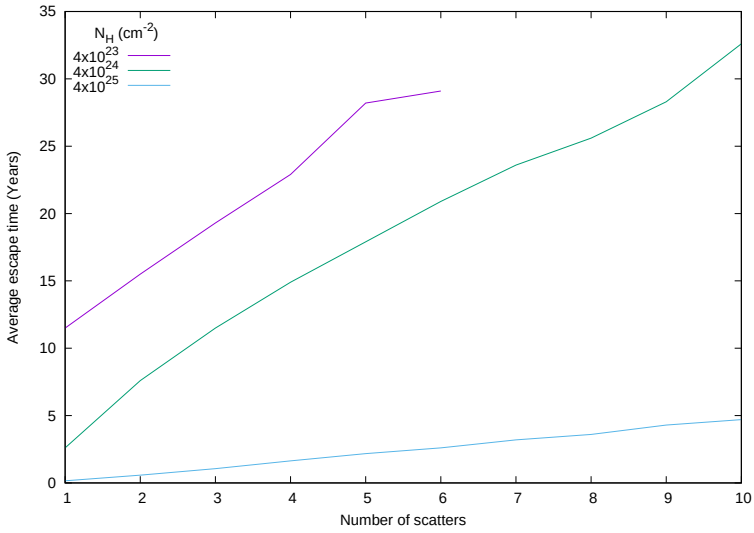


Figure 3.21: Average escape time from a 2 pc diameter cloud and number of times scattered. Showing three column density values.

3.7.2 Spectral Shape

Figure 3.22 shows the spectral output from the code at different time periods, as well as a summed spectrum of those different periods. The summed spectrum represents the usual output of the code, as has been presented up to this point. All of the time periods from 0 years and up show the same low energy shape and flux. However, the time period from 5-100 years shows a markedly different low energy shape, because all of the low energy photons will leave the cloud very quickly. The high energy shape and flux has a large discrepancy from the summed spectrum in the early years, but as time increases the high energy will approach the fully summed spectrum as more and more of the multiply scattering photons get released. There is a large discrepancy in continuum shape between 10 and 20 keV for the different time periods. This discrepancy is caused by the continued release of multiply scattered photons as time increases. In order to avoid potential issues with the fitting procedure, the energy range 10-20 keV will have to be ignored for all fittings. The high energy flux will not be possible to determine correctly without further research into time dependent effects. As a result, high energy flux values will also have to be ignored in the fitting results. However, as can be seen in Figure 3.22 the slope of the high energy spectrum does not change depending on the time period, thus it can still be used to determine the photon index.

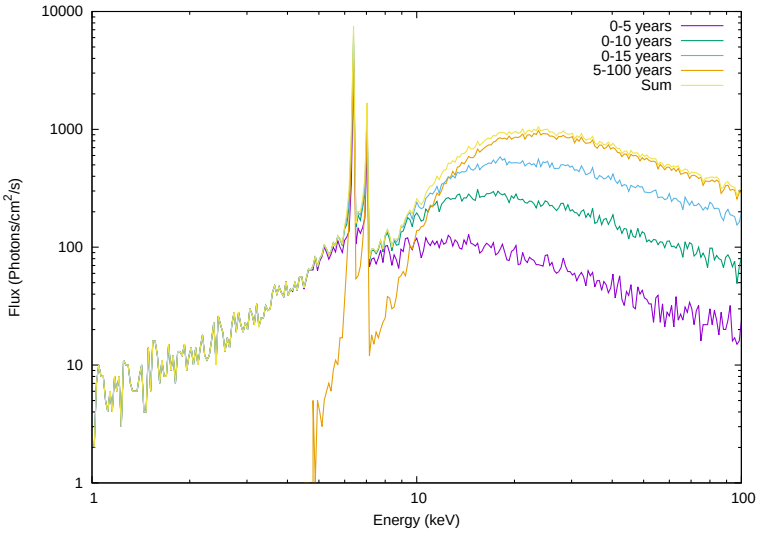


Figure 3.22: Comparison of the output spectra of a Compton thick cloud, for various time periods. Where *Sum* is the output spectra as presented thus far.

Chapter 4

Data Fitting

The primary goal of this thesis is to fit the simulated X-ray spectra to real observational data. This section will cover the method of fitting and the fitting software used, followed by a summary of XSPEC table models and specifics for the table models used in this project. Finally, a brief summary of the observational data will be presented.

4.1 XSPEC

In order to perform the X-ray spectral fitting this project made use of specialised software known as XSPEC[1] (Arnaud, 1996). XSPEC is a command line driven, interactive program. It is detector-independent, meaning it can be used with any kind of spectrometer. XSPEC is essentially the de facto X-ray fitting software worldwide and is used in the analysis of data from many instruments including HEAO-1 A2, EXOSAT, and Ginga to ROSAT, Chandra, XMM-Newton, INTEGRAL, Swift and Suzaku. How XSPEC performs its fitting is covered in Section 4.1.1 and the XSPEC table models, which were used to import the Monte Carlo code into XSPEC are covered in Section 4.1.2.

[1] https://heasarc.gsfc.nasa.gov/xanadu/xspec/

4.1.1 Fitting Methods

The extensive details of the XSPEC fitting process are covered in the XSPEC manual[1]. An abridged version will be discussed here.

A spectrum obtained from the spectrometer is not actually a spectrum, it is photon counts (C) within specific instrument channels (I), $C(I)$, with the observed spectrum being related to the real spectrum $f(E)$ by equation (4.1).

$$C(I) = \int_0^\infty f(E)R(I,E)dE, \tag{4.1}$$

where $R(I,E)$ is the instrument response. $R(I,E)$ is proportional to the probability that an incoming photon of energy E will be detected in channel I. Unfortunately, inverting equation (4.1) and solving for $f(E)$ is not usually possible. Instead, an alternative method of determining $f(E)$ is used. This method involves defining a model spectrum $m(E)$, which has a small number of variable parameters, p1, p2 etc, and fitting it to the data. For every model spectrum $m(E)$ a predicted count spectrum ($C_p(I)$) is calculated. This predicted count spectrum is then compared to the observed count spectrum $C(I)$. A fit statistic is calculated by comparing the two and is used to determine if the model spectrum is a good fit for the observed spectrum. The fitting procedure uses a least squares fitting algorithm, which is a modified version of the Levenberg-Marquardt algorithm. This modified Levenberg-Marquardt algorithm is appropriated from Bevington (1969) according to the XSPEC manual.

XSPEC will determine the best fit statistic by varying the parameters of the model until the best fit is found. The parameters that give this best fit are known as the *best fit parameters* and the model spectrum that results from these parameters $m_b(E)$ is the *best-fit model*. There are a few best fit statistics that can be used; the most common and the one used in this project is the chi-square χ^2 fit statistic, which is defined by equation (4.2).

$$\chi^2 = \sum \frac{(C(I) - C_p(I))^2}{(\sigma(I))^2}, \tag{4.2}$$

where $\sigma(I)$ is the error for each instrument channel I, $\sigma(I)$ being estimated as $\sqrt{C(I)}$. After finding the *best-fit model* there is still the question of the quality of the fit, for which there are two considerations. The first is the *goodness-of-fit*, which is a measure of the confidence that the observed counts $C(I)$ can be

[1] https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/manual.html

Confidence	Parameters		
	1	2	3
0.68	1.0	2.3	3.50
0.90	2.71	4.61	6.25
0.99	6.63	9.21	11.30

Table 4.1: Confidence interval values

produced by the best fit model $m_b(E)$. For the χ^2 fit statistic the *goodness-of-fit* is known as the reduced χ^2 and is given by equation (4.3).

$$\chi^2_{red} = \frac{\chi^2}{dof}, \tag{4.3}$$

where dof is the *degrees of freedom*. The dof is the numbers of instrument channels minus the number of model parameters. A general rule is that the χ^2_{red} should be approximately equal to one. A value greater than one indicates a poor fit, while a value less than one indicates the errors in the data have been over-estimated. An important caveat is that there can be more than one model that will pass the *goodness-of-fit* test quite well, in which case the correct model must be left up to scientific judgement. A second consideration is confidence in the value of the predicted parameters; this is known as the *confidence interval* of the parameter. The confidence interval is found by varying the value of the parameter until the χ^2 exceeds a certain value above the minimum (the best-fit value). The values by which the χ^2 is allowed to increase depend on the confidence level required and the number of parameters. Common values are shown in Table 4.1 (Avni, 1976).

4.1.2 Table Models

XSPEC already includes a very large selection of models for data fitting, details of which can be found here[1]. However, for this project a way was needed to define a specific user generated model for use in XSPEC. There are two main methods of getting user generated models into XSPEC. The first is defining the model as a simple formula, the second is generating what is known as a table model. A table model is a model that is defined by a grid of input spectra, with the elements of

[1] https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/XspecModels.html

the grid representing different values of the input parameters. For example, a two parameter model (α, θ) where each parameter has three potential values (1,2,3) will be organised in a grid such as that shown in Table 4.2. For each point on the grid a different spectrum is calculated using the parameter values at that point, with the correct spectrum for any particular parameter values being calculated by linear interpolation on the grid.

(α_1, θ_1)	(α_1, θ_2)	(α_1, θ_3)
(α_2, θ_1)	(α_2, θ_2)	(α_2, θ_3)
(α_3, θ_1)	(α_3, θ_2)	(α_3, θ_3)

Table 4.2: Table model grid layout for two parameter model, each parameter having 3 potential values.

There are three types of model available in XSPEC: the additive model, the multiplicative model and the exponential model. The exponential model is a special case that is primarily intended for absorption models, while the multiplicative models are models that modify additive models by some factor. Additive models are the default type of model and are the type of model used in this project. The file structure for these models is a FITS file, with a specific file structure which is described here[1].

There were two primary table models created for the data fitting in this project. As they were primarily intended for fitting to Sgr B2 they were designed as a uniform density and a Gaussian distributed non-uniform density model. Other models were used at times, but their descriptions would be superfluous. The uniform density model used a sphere of radius 2 pc, situated a distance of 25 pc from the source. Of course, given the nature of the code, cloud size and distance will not affect the output spectra unless taken to extreme levels. This is because all photons are released toward the cloud, a noticeable effect will only be present if the cloud is so large as to be reflecting photons from very divergent angles. The three parameters of the model are the line of sight angular position θ, the photon index of the incident spectrum Γ and the column density of the cloud N_H. The column density being directly dependent on the density ρ and radius of the cloud. These three parameters were included over the ranges shown in Ta-

[1] http://heasarc.gsfc.nasa.gov/docs/heasarc/ofwg/docs/general/ogip_92_009/ogip_92_009.html

ble 4.3. The N_H parameter steps are not evenly spaced, but are logarithmically spaced, while the angle parameter steps are evenly spaced in cosine.

Parameter	Start	End	Steps
θ	0	π	21
Γ	1.4	3.0	17
N_H (cm^{-2})	1.049×10^{22}	5.184×10^{25}	20

Table 4.3: Parameter ranges for uniform density table model.

The Gaussian table model took its parameters from Protheroe et al. (2008), who found Sgr B2 to have a Gaussian density distribution with the following parameters: Radius = 12 pc, Central density $\rho_0 = 1.2 \times 10^5$ cm^{-3}, which is a total $N_H = 2.5 \times 10^{24}$ cm^{-2}, and finally a standard deviation $\sigma = 2.75$ pc. The table model follows these parameters. The parameter ranges are shown in Table 4.4, the central density ρ_0 parameter steps are spaced so that the resultant N_H is still logarithmically spaced.

Parameter	Start	End	Steps
θ	0	π	21
Γ	1.4	3.0	17
ρ_0 (cm^{-3})	1.0×10^4	1.0×10^6	20

Table 4.4: Parameter ranges for Gaussian density table model.

4.2 Data Choice and Analysis

The data used needed to have good statistics in order to perform an accurate fitting. There was also a need to have broadband data. In order to constrain the photon index, data above $\approx$ 20 keV was required. To that end, there were 3 primary data sets used for the spectral fitting: A *Chandra* observation from 2000, an *XMM-Newton* observation from 2004 and for the high energy data, an *INTEGRAL* observation from 2004. The data was provided by colleagues in France, thus data reduction was not performed by the author. However, a short summary of the data will be given here and sources provided for a detailed description of the data reduction.

4.2.1 Chandra

For the *Chandra* data see Walls et al. (2016), where the following description is given.

> *"Sgr B2 was observed by Chandra for 100 ks in July 2000 when it was very bright. We have analysed the corresponding dataset (ObsID 944) using* `ciao` *v 4.3. Event lists were cleaned for high background periods yielding effective exposures of respectively 97 ks.*
>
> *Images were extracted in the energy ranges 3-6 keV and 6.3-5.6 keV, which contains most of the 6.4 keV Fe Kα line. To produce point source free diffuse emission maps, we performed source detection in the 3-6 keV energy range using the* `wavdetect` *task and removed the regions surrounding the source and filled the hole using the count rate estimated locally taking into account Poisson random fluctuations with the task* `dmfilth`*. Exposure maps were then produced in each energy interval assuming an absorbed power law spectral shape to weight the different energy contributions to the total exposure. We used the absorption to the Galactic centre ($N_H = 7 \times 10^{22} cm^{-2}$) and a power law index of 2. The background was estimated using blank field data provided in the calibration database (CALDB) and normalized to the images using the high energy count-rate. The point source cleaned images were then corrected for the background and divided by the corresponding exposure. Finally, we smoothed to produce flux images of the diffuse emission, see Figure 4.1.*
>
> *We have extracted the spectrum from a region of 3.2 arcmin radius centred on the peak of the column density at l= 0.66° and b= -0.03°. Spectral extraction and instrument responses were generated using the* `specextract` *task. Because of the intensity of the local astrophysical background and its variations in the field of view, we used blank sky observations provided by the CXC to estimate the background contribution in each region."*

Figure 4.1 shows the core of Sgr B2 in 6.4 keV iron line emission. The extraction region used for the *Chandra* and *XMM-Newton* data is shown as the white circle.

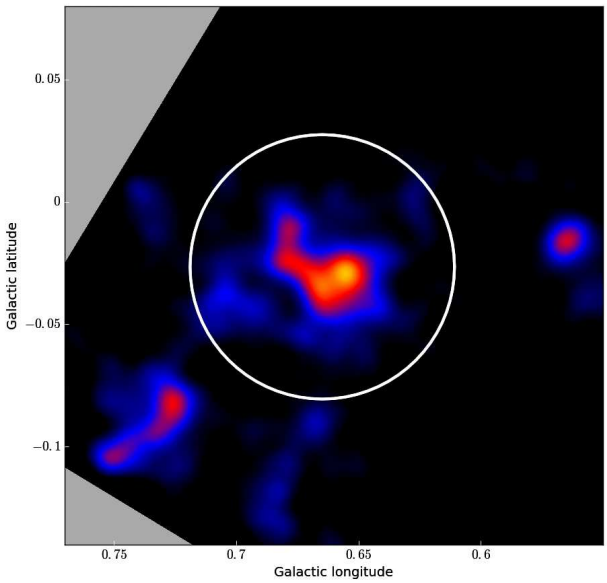


Figure 4.1: 6.4 keV line flux image obtained with *Chandra* in 2000. The bright core of Sgr B2 in the centre, surrounded by more diffuse emission. The *Chandra* and *XMM-Newton* spectra have been extracted from a region (white circle) of 3.2′ radius centred on the core of Sgr B2. From Walls et al. (2016).

4.2.2 XMM-Newton

For the *XMM-Newton* data again see (Walls et al., 2016), where the following description is given.

> *"We analyzed the XMM-Newton data taken in September 2004 during a dedicated 50 ks exposure of the Sgr B2 molecular cloud (ObsID 0203930101). Because of significant contamination by flare events we did not use the EPIC/PN data. After cleaning the high background time intervals, the resulting exposure time with the EPIC/MOS cameras is 40 ks. We extracted the spectra from the EPIC/MOS instruments using the Extended Source Analysis Software (ESAS) (Snowden et al., 2008) distributed with version 12.0.1 of the XMM-Newton Science Analysis Software."*
>
> *We produced calibrated and filtered event files with the tasks emchain and mos-filter, in order to exclude the time intervals affected by soft proton contamination. The spectra were extracted from the same region as the Chandra using the ESAS mos-spectra scripts and were*

re-binned to have at least 30 counts in each bin to apply chi-square statistics. The background was obtained from the filter wheel closed archival observations provided within the ESAS database. It provides an estimate of the quiescent component of the EPIC internal particle background.

The spectral extraction region radius of 3.2 arcmin (7.5 pc at GC distance) was chosen because it provides a full coverage of the cloud with 99% of the mass enclosed, according to the density profile obtained by Protheroe et al. (2008). It was also chosen to provide a relatively good match with the *INTEGRAL* IBIS/Isgri PSF (12′ FHWM) and to ensure a stable comparison between the soft and hard X-ray datasets (Walls et al., 2016).

4.2.3 INTEGRAL

Finally, for the *INTEGRAL* data please see Terrier et al. (2010). They cover observations taken during the years 2003 and 2004, roughly contemporaneous with the *XMM-Newton* data. This paper contains details on data treatment, calibration and spectrum extraction. The *INTEGRAL* source was found perfectly coincident with the Sgr B2 core region visible in Figure 4.1. The X-ray emission and the hard X-ray spectrum seen with *INTEGRAL* are therefore almost certainly from the same source.

Chapter 5

Results

Chapter 2 described the Monte Carlo code created for this project. Section 4.1 described the XSPEC fitting process. Section 4.2 presented the observational data that will be used in the fitting process. In the forthcoming chapter, the results of fitting the XSPEC table models to the observational data will be presented, beginning with individual fits of the data, followed by the simultaneous fits of the data as presented in Walls et al. (2016).

5.1 Specifics of Sgr B2 as Related to Data Fitting

The code is designed to simulate an isolated externally illuminated molecular cloud, however in real world situations there are often more factors involved, and Sgr B2 is not an exception. As briefly mentioned in Chapter 1, there is significant warm plasma emission from the Galactic centre area (Ryu et al., 2009; Muno et al., 2004); this emission is seen in the Sgr B2 data. In order to account for this emission the data must be fit, not just with the table model, but also with a plasma model. The warm plasma model used is the *apec* model (Astrophysical Plasma Emission Code) (Smith et al., 2001) that comes with XSPEC. Details of the *apec* model can be found in the XSPEC manual[1].

Thermal emission in the GC is usually described by a double plasma model with typical temperatures on the order of 1 keV and 7 keV (Koyama et al., 1989; Tanaka et al., 2000). The origin of the 1 keV plasma is believed to by heating by supernova in the GC, while the origin of 7 keV plasma remains unknown (Koyama et al., 2007). There is also a small amount of absorption that takes place along

[1] https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/XSmodelApec.html

the line of sight from Sgr B2 to Earth, this absorption is included via the XSPEC model *wabs*, information for which can again be found in the XSPEC manual[1].

Thus, the final model that the data has been fit with is as follows:

$$wabs(apec+apec+Table\_Model)$$

Unfortunately, the fitting of the data was unable to properly constrain the plasma temperatures, as a result the plasma temperatures needed to be frozen at reasonable values. In the central regions of the Galaxy, Muno et al. (2004) found temperature values of 0.7-0.9 keV for the cool component and 6-8 keV for the hot. Numerous studies with *Suzaku* have shown the soft plasma to have roughly a temperature of 1 keV in many regions in the CMZ (Nobukawa et al., 2008; Mori et al., 2008), while Koyama et al. (2007) found that the hot plasma is well described by a plasma with temperature 6.5 keV and solar metallicity. In the Sgr B2 region, Ryu et al. (2009) performed a fitting with a 3 *apec* warm plasma model. The first of these models only contributes below 2 keV and is thought to be local. This energy range was excluded from the fitting and thus this contribution is neglected. The second, the soft plasma temperature, was fitted and they found kT = 0.7 keV. The third, the hot plasma temperature was fixed at 6.5 keV. For this project the fitting follows the same approach as Ryu et al. (2009), but with the soft component temperature fixed to 1 keV as a compromise. Note that taking into account the absorption to the GC the main effect of a change of 0.7 - 1 keV above 2 keV is mainly in the relative intensities of the plasma lines. Thus, the plasma temperatures of the *apec* models are frozen at 1 keV and 6.5 keV respectively. All errors included in the following results are calculated by XSPEC and represent a 90% confidence value. The method of their calculation is covered in Section 5.5.2.

[1] https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/XSmodelWabs.html

5.2 Individual Fits

The data being used, as covered in Chapter 4, can be fit with the table models in XSPEC either individually, or simultaneously. Fitting the data individually does not provide as good a constraint on the relative parameters as a simultaneous fitting, but it is useful in gauging the accuracy of the simultaneous fit. Thus, the results of the individual fits are covered in this section.

5.2.1 Chandra Fits

The Chandra data (Section 4.2.1) was fit with both the uniform density and the Gaussian density table models, using the energy range 2 - 7.5 keV.

5.2.1.1 Chandra Uniform Density Fit

The first presented fitting is the *Chandra* data fit with just the table model, no plasma models have been included. The $\chi^2_{red} = 1.03$ of the fit is actually very good, but the parameters become very poorly constrained. However, the absolute quality of the fit is not always the final word involved in finding the best result.

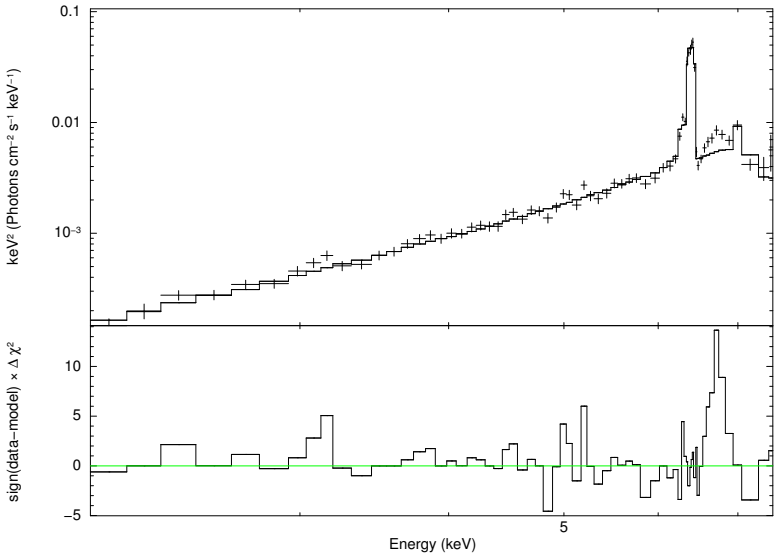


Figure 5.1: Best uniform density fit to *Chandra* data, without the double plasma model.

For example, Figure 5.1 shows the Chandra data fit with just the uniform density table model, compared to Figure 5.2 which shows the full model with the *apec* plasma model. Specifically notice the peak at 6.7 keV situated between the K_α and K_β emission lines. This peak is the Fe XXV emission mentioned

in Chapter 1 and is strictly from the plasma emission, specifically the 6.5 keV plasma. Consequently, although the parameters become less constrained when using the plasma model, it shows that it is needed nevertheless, particularly because the fitted parameters change with the addition of the plasma model.

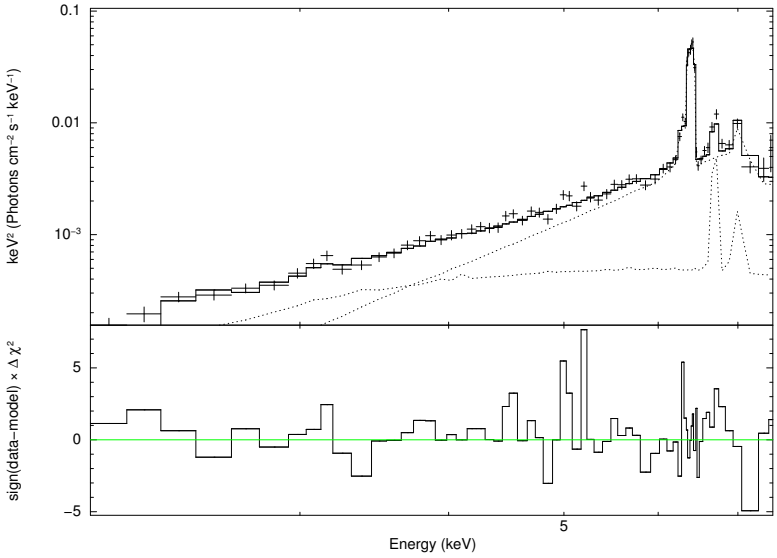


Figure 5.2: Best uniform density fit to *Chandra* data.

The results of the full uniform density model fit to the *Chandra* data can be seen in Figure 5.2 and Table 5.1. While the errors are quite high, which speaks to the parameters being poorly constrained, the $\chi^2_{red} = 1.04$ can be interpreted as a sign that the data is indeed of an X-ray reflection spectra with a Compton thick N_H. The error on the photon index appears to be quite low in the negative. This is a result of the fitting hitting the hard lower limit in the model which is 1.4; in actuality the photon index parameter is not well constrained.

θ	N_H (10^{24} cm^{-2})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
58^{+27}_{-30}	$5.04^{+1.56}_{-3.56}$	$1.41^{+0.46}_{-0.01}$	$5.55^{+1.57}_{-1.5}$	1 & 6.5	365 (352)

Table 5.1: Best fit parameters for uniform density table model to the *Chandra* data.

5.2.1.2 Chandra Gaussian Density Fit

The Gaussian model fit can be seen in Figure 5.3 and Table 5.2. The same effect from Section 5.2.1.1 is also apparent with the Gaussian fitting, where the parameters are better constrained and more reasonable without taking the plasma emission into account.

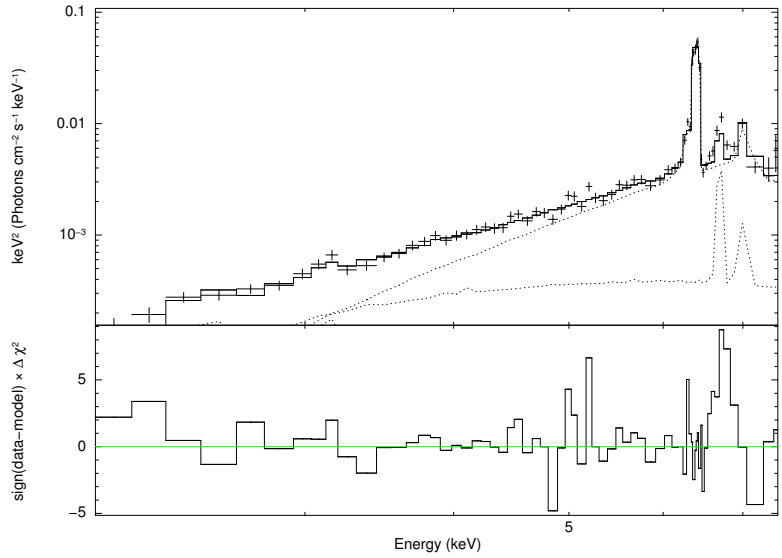


Figure 5.3: Best Gaussian density fit to the *Chandra* data.

The Gaussian fit in general is slightly worse than the uniform, but the $\chi^2_{red} = 1.07$ is still very good, and the density parameter fits quite well. However, the angle parameter is poorly constrained and the photon index is still hitting the hard lower limit implying the data is completely insensitive to the photon index parameter. A photon index this low cannot be discounted from this fitting alone, that said, it is not a reasonable value based on previous estimates (see Chapter 6).

θ	ρ_0 (10^5 cm^{-3})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
82^{+19}_{-15}	$0.89^{+0.49}_{-0.13}$	$1.4^{+0.13}$	$5.78^{+1.6}_{-0.78}$	1 & 6.5	378 (352)

Table 5.2: Best fit parameters for Gaussian density table model to the *Chandra* data.

5.2.2 XMM-Newton Fits

The *XMM-Newton* fitting is done in the energy range 2 - 9 keV. There are two actual data sets that are used as one; they are the data from both MOS cameras on *XMM-Newton*. As stated in Section 4.2.2, the PN data is not included due to significant contamination from flaring events. The *XMM-Newton* fit, in contrast to the *Chandra* fit, improves with the addition of the plasma emission and so a comparison between plasma and non-plasma fittings will not be shown in this section as before.

5.2.2.1 XMM-Newton Uniform Density Fit

The *XMM-Newton* uniform density fit is shown in Figure 5.4 & Table 5.3. The fit quality $\chi^2_{red} = 1.01$ is good, and the parameters are better constrained than in the *Chandra* fit. Although the photon index parameter appears to be reasonably constrained by the model, it is clear by the fact it keeps hitting the hard lower limit, that this low energy data is not capable of discerning the photon index by itself. For this reason, high energy *INTEGRAL* data is also used. The fit being shown in Section 5.2.3.

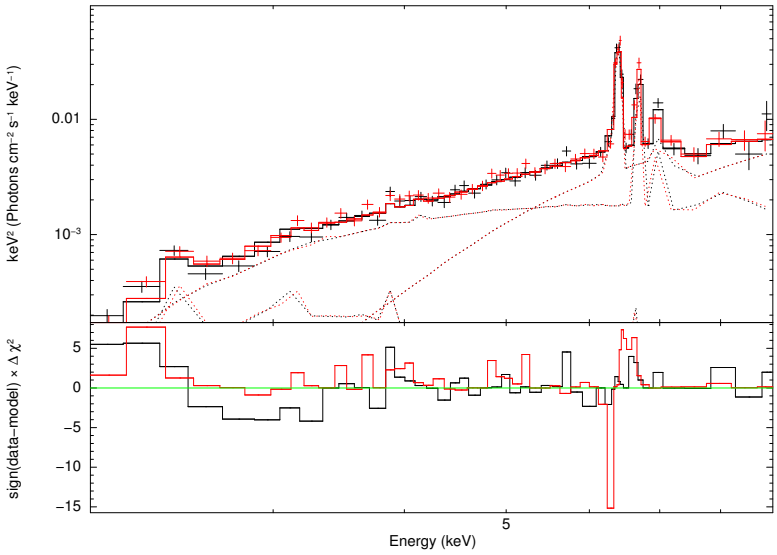


Figure 5.4: Best uniform density fit to the *XMM-Newton* data.

θ	N_H (10^{24} cm^{-2})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
$60^{+9.3}_{-7.8}$	$0.83^{+0.21}_{-0.13}$	$1.40^{+0.4}_{-0.0}$	$7.95^{+0.75}_{-0.74}$	1 & 6.5	688 (676)

Table 5.3: Best fit parameters for uniform density table model to *XMM-Newton* data.

5.2.2.2 XMM-Newton Gaussian Density Fit

The *XMM-Newton* Gaussian fit is shown in Figure 5.5 & Table 5.4 and it is, as with the *Chandra* data, a reasonable if slightly worse fit, that also constrains the parameters well.

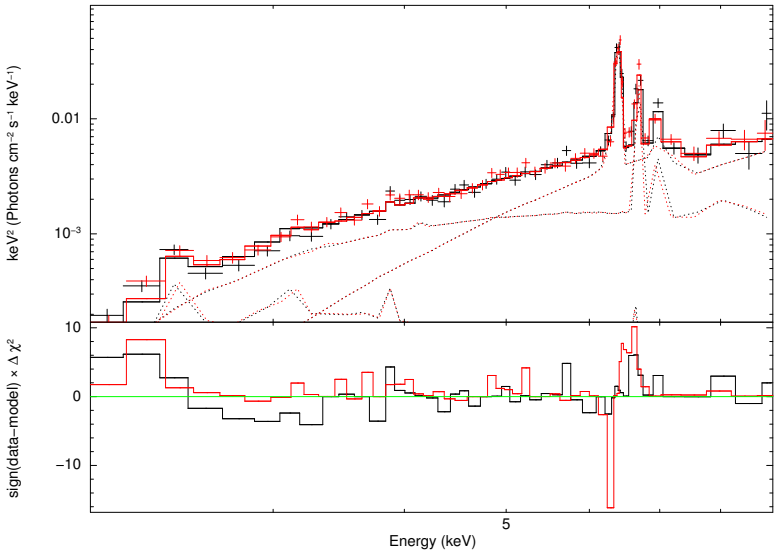


Figure 5.5: Best Gaussian density fit to *XMM-Newton* data.

θ	ρ_0 (10^5 cm^{-3})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
52^{+13}_{-14}	$0.72^{+0.68}_{-0.17}$	$1.40^{+0.2}_{-0.0}$	$7.82^{+0.97}_{-0.88}$	1 & 6.5	706 (676)

Table 5.4: Best fit parameters for Gaussian density table model to *XMM-Newton* data.

5.2.3 INTEGRAL Fits

The low energy spectrum is not able to constrain the photon index parameter very well, but the high energy can potentially be used to constrain this parameter. The *INTEGRAL* data was fit in the energy range 20.0 - 70.0 keV. These *INTEGRAL* fits are very poor in general. As shown in Chapter 3, the high energy spectrum does not change much with angle or N_H, so this poor fit is to be expected. As a result, the fitting done for the *INTEGRAL* data alone has had both angle and N_H frozen to reasonable values with just the photon index left variable. Unfortunately, even with these parameters frozen, the error calculation by XSPEC is not possible given the very high $\chi^2_{red} > 2.0$. Although both the uniform density and the Gaussian density fits are poor, they present almost identical results. Again, this is to be expected from Chapter 3. As a result, only the uniform density fit is presented here and can be seen in Figure 5.6 and Table 5.5.

5.2.3.1 INTEGRAL Uniform Density Fit

θ	N_H (10^{24} cm^{-2})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
70	2.0	2.25	7	1 & 6.5	12 (4)

Table 5.5: Best fit parameters for uniform density table model to *INTEGRAL* data. θ, N_H and wabs were all fixed. Γ was left free.

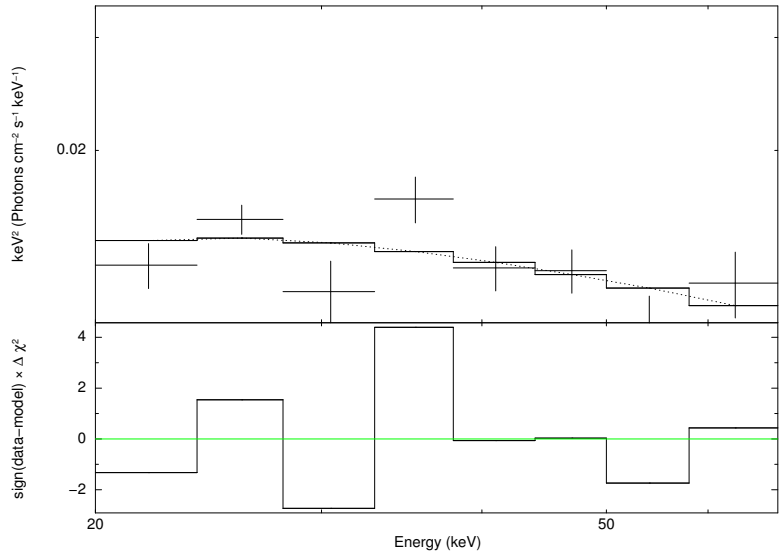


Figure 5.6: Best uniform density fit to *INTEGRAL* data.

5.3 Simultaneous Fits

Fitting the data simultaneously provides a far better constraint on the parameters. There are specific considerations to be taken into account when fitting data from different time epochs simultaneously. Generally, they do not affect the final results, as such the results are presented straight-forwardly here. There are further discussions on these considerations in Section 6.5.

5.3.1 Uniform Density Fit

The results of the uniform density fit are shown in Figure 5.7 and Table 5.6. As expected, all of the parameters are better constrained than in the individual fits. Most importantly, the photon index is not only well constrained but has a far more reasonable value. The fit quality is good $\chi^2_{red} = 1.18$. Contour plots of this fit can be seen in Appendix A.1.

θ	N_H (10^{24} cm^{-2})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
64^{+8}_{-7}	$2.13^{+0.28}_{-0.35}$	$2.16^{+0.12}_{-0.16}$	$6.31^{+0.9}_{-0.89}$	1 & 6.5	731 (618)

Table 5.6: Best simultaneous fit for the uniform density table model.

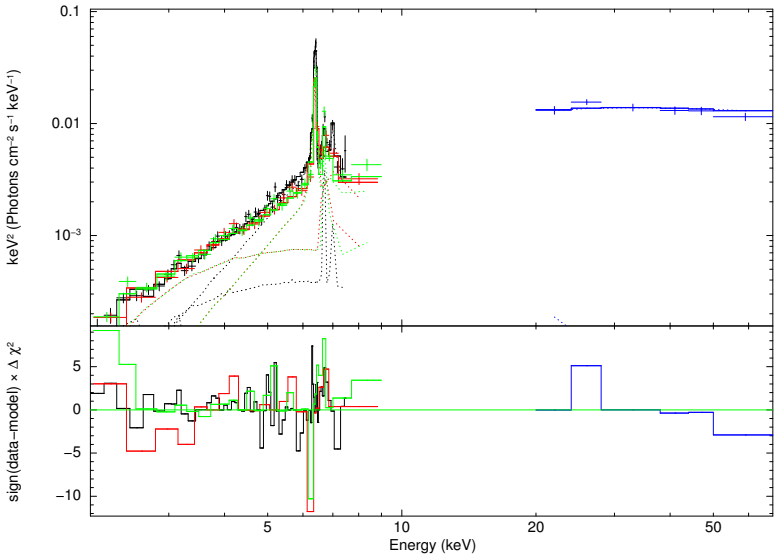


Figure 5.7: Best simultaneous fit with the uniform density table model.

5.3.2 Gaussian Density Fit

The results of the Gaussian density fit are shown in Figure 5.8 and Table 5.7. The central density parameter is given in terms of N_H (cloud radius 12pc, standard deviation $\sigma = 2.75$pc) to aid comparison with the uniform density fit.

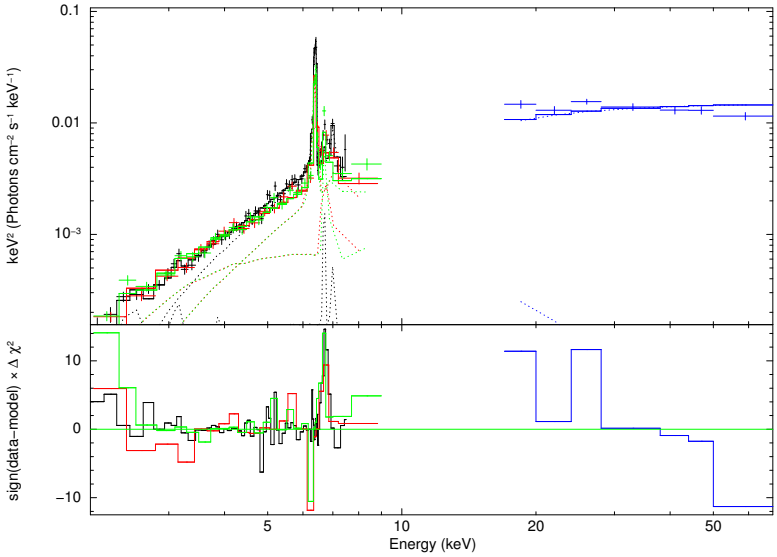


Figure 5.8: Best simultaneous fit with the Gaussian density table model.

The fit is not as good as the uniform density fit, $\chi^2_{red} = 1.33$. While still borderline acceptable, it is slightly high given the number of degrees of freedom. The parameters are also slightly less constrained than in the uniform density fit. Contour plots of this fit can be seen in Appendix A.2.

θ	N_H (10^{24} cm^{-2})	Γ	wabs N_H (10^{22} cm^{-2})	kT 1 & 2 (keV)	$\chi^2(dof)$
89^{+10}_{-11}	$2.5^{+0.43}_{-0.66}$	$1.83^{+0.06}_{-0.11}$	$6.79^{+0.78}_{-0.84}$	1 & 6.5	820 (618)

Table 5.7: Best simultaneous fit for the Gaussian density table model.

5.4 Luminosity

The observed flux and the luminosity incident on Sgr B2, as derived from the results presented in Section 5.3, can be seen in tables 5.8 and 5.9. The method of luminosity calculation is presented in Section 5.4.1. As discussed in Section 3.7, the time dependence makes it impossible to fit or to make use of normalisation values for energies above 10 keV. For this reason, flux and luminosity calculations are only carried out in the energy range 2 - 10 keV.

Observation	$Flux_{2-10}$ ($10^{-12} erg/cm^2/s$)	L_{2-10} $10^{39}(erg/s)$
Chandra	$7.34^{+0.12}_{-0.17}$	$1.57^{+0.2}_{-0.4}$
XMM-Newton	$5.29^{+0.11}_{-0.11}$	$1.13^{+0.2}_{-0.2}$

Table 5.8: Fluxes and associated luminosities for *Chandra* and *XMM-Newton* data, obtained from the uniform density fit.

Observation	$Flux_{2-10}$ ($10^{-12} erg/cm^2/s$)	L_{2-10} $10^{39}(erg/s)$
Chandra	$6.97^{+3.21}_{-0.11}$	$2.03^{+1.94}_{-0.4}$
XMM-Newton	$5.11^{+1.7}_{-0.15}$	$1.48^{+0.5}_{-0.2}$

Table 5.9: Fluxes and associated luminosities for *Chandra* and *XMM-Newton* data, obtained from the Gaussian density fit.

5.4.1 Luminosity Calculation

The incident luminosity was calculated by the following method. From the output of the code, there are two important values: the total flux into the cloud and the total flux observed out of the cloud in a particular energy range. These are F_{in} and F_{out} respectively. With their respective physical representations being that flux from Sgr A* which impacts Sgr B2, and that flux which is emitted from Sgr B2 and observed.

From XSPEC, there is the flux in units of $(erg/cm^2/s)$, which is F_{Xspec}. From the values of F_{in} and F_{out}, the *Cloud Ratio* can be calculated. The *Cloud Ratio* is given by equation (5.1). This is, in abstract terms, the effect the cloud has on the incident flux.

$$CR = \frac{F_{in}}{F_{out}}, \tag{5.1}$$

Also needed is the XSPEC *Factor*, which is given by equation (5.2).

$$XF = S_C \times (\frac{D_R}{D_C})^2, \tag{5.2}$$

where $S_C = \pi \times R^2$ is the surface area of the observer, with R the radius of the observer in the code. D_R and D_C are respectively the distance from the cloud to the observer in reality and in the code. The XSPEC *Factor* gives the relationship between the XSPEC flux F_{Xspec} in $(erg/cm^2/s)$ and the total flux leaving the cloud in the direction of the observer in (erg/s). The XSPEC *Factor* is then used as per equation (5.3).

$$F_o = F_{Xspec} \times XF, \tag{5.3}$$

where F_o is the total flux leaving the cloud in the direction of the observer. Then to be calculated is that flux which entered the cloud; this is related to F_o by the *Cloud Ratio* and so is given by equation (5.4).

$$F_{Real} = F_o \times CR, \tag{5.4}$$

where F_{Real} is the real flux into the cloud. This flux is then converted to the luminosity leaving the source by equation (5.5).

$$L = \frac{4\pi \times F_{REAL}}{\Omega_C}, \tag{5.5}$$

where Ω_c is the solid angle of the cloud as seen by the source, given by equation (5.6).

$$\Omega_C = 2\pi(\frac{R_C - D_C}{\sqrt{D_C^2 + R_C^2}}), \tag{5.6}$$

where R_C is the radius of the cloud.

5.5 Errors

There are several ways in which errors can crop up in this kind of fitting. The first is errors inherent in the Monte Carlo approach; which are covered in Section 5.5.1. The second is statistical errors in the fitting procedure, which are the errors XSPEC calculates; they are covered in Section 5.5.2. Finally, the third are errors that can be introduced by the human element doing the fitting. These errors are not possible to quantify, but can lead to incorrect results. This topic is briefly covered in Section 5.5.3.

5.5.1 Monte Carlo

Monte Carlo calculations of this sort are a Poisson process. Sampling of the sort discussed in Chapter 2 will be free of errors assuming it is being sampled in high enough numbers, and the PRNG being utilized is sufficiently random. However, the binning of photons is different. Just as in a CCD, binned photons will follow a Poisson distribution when dealing with low counts, and as per the central limit theorem, a Normal Distribution when dealing with high counts. The error in each bin is given by equation (5.7) (Wood et al., 1996).

$$\Delta F = \frac{F}{\sqrt{N_i}}, \tag{5.7}$$

where F is flux and N_i is the number of photons in the bin. For the most part, this error is insignificant. As counts start to get low the error can become significant, for example, a particular reflected spectra may have only 8 counts in the low energy heavily absorbed region around 1-2 keV. This is an error of 2.8, which is far from insignificant. However, not only do counts this low occur extremely infrequently (in fact only in the 0 degree case, see Chapter 3), but the effect on the fitting process is negligible, as relative to the shape and normalisation of the total spectra, it is entirely insignificant.

5.5.1.1 Pseudo Random Number Generators

As already mentioned in Section 2.2.1.1, there are many available algorithms to produce pseudo random numbers (prng). The original idea was to use the c++11 implementation of a Mersenne twister algorithm, which is a very advanced prng and would be more than adequate for this project. However, the HPC (High Performance Computer) being used in Ichec was unable to compile c++11 code, thus with the possible exception of implementing my own algorithm or using an external library, I was forced to use the old C rand() function. Unfortunately, the C rand() function is known to be one of the weaker prngs available in C++. However, given the extremely large number of photons being simulated, it is unlikely that using rand() would be a problem. In order to ensure this is the case, the code output produced by both prng's was compared for a number of different input parameters, and with total photons counts reduced from the standard (10^9) by a factor of 100 to account for low count, noisy situations. There were no statistically relevant differences between the outputs of the code in any case. Thus, the rand() function was deemed to be adequate for this project.

5.5.2 XSPEC Error Calculation

As already discussed in Section 4.1.1, errors in XSPEC are calculated by a relatively straightforward method. When the error for a particular parameter is requested, XSPEC will vary the parameter (other non-frozen parameters are also allowed to vary) within its given hard limits until the value of the fit statistic (in the case of this project, the χ^2 value) is equal to the value of the best fit statistic last determined by a regular fitting, plus the *delta fit statistic*. The *delta fit statistic* is kept within an absolute tolerance value, which for this project is kept at its default value of 0.01. *delta fit statistic* values were shown in Table 4.1. All the results presented so have used the default confidence level of 90%.

5.5.3 Human Error

As was mentioned a few times throughout Chapter 5, although a fitting can show what on the surface seems to be a very good result (as in a χ^2_{red} that is very close to unity) the reality is not always so. The two main examples from the fitting performed in this project are as follows.

In Section 5.2.1.1 where the individual Chandra data fits are presented, the fitting performed without the addition of the plasma model was on the surface a very good fit (well constrained parameters and χ^2_{red} very close to unity). However, when the spectrum of the data was looked at closely, it was apparent that the plasma emission was significant and could not be excluded, particularly because its addition did change the results.

The second example was when fitting just the low energy data (Sections 5.2.1.1 - 5.2.2.2) the photon index parameter appeared to be well constrained and the χ^2_{red} was good. However, the fact that the fitting was stuck to the hard lower limit of the parameter, combined with the value of 1.4, which is at odds with the literature (Terrier et al., 2010; Ponti et al., 2013), meant that in actuality the fitting was not correctly finding the photon index parameter. The later addition of the higher energy *INTEGRAL* data solved this problem.

These two examples demonstrate how fitting of the type performed in this project is not necessarily an exact science and much can be left up to the judgement of the person performing the fitting. The conclusion being that scientific judgement is just as important in interpreting the fitting results as the fit statistics. Indeed, as the number of degrees of freedom goes up in any particular fitting, it is important to ensure the fitting procedure is not stuck in a local minimum, this also requires human judgement.

Chapter 6

Discussion

In this section, the results obtained will be discussed and compared to those results from the literature, which are relevant. There are several further points that also require mention and will be discussed in this section. They are in order: the iron abundance, the density profile, and the impact of the time dependence.

6.1 Summary of Results

The individual fits, although useful in broadly constraining the parameters, are not as accurate as the simultaneous fits, in particular not constraining the photon index parameter very well. The simultaneous fit contains a lot more data points for model fitting and so provides a far better constraint on the interested parameters. For this reason the simultaneous fit results are considered to be the better results and only they will be discussed further. The results obtained are as follows.

For the uniform density, the fit quality is quite good with a $\chi^2_{red} = 1.18\,(618)$. Sgr B2 is found to have a Compton thick dense core with a hydrogen column density $N_H = 2.13^{+0.28}_{-0.35} \times 10^{24}$ cm^{-2}. The photon index of the incident spectrum was found to be $\Gamma = 2.16^{+0.12}_{-0.16}$. The fitted angular position was $\theta = 64^{+8}_{-7}$ degrees, which implies a distance from Sgr $A^\star$ to Sgr B2 of 111^{+8}_{-6} pc, assuming a projected distance of 100 pc. Taking a distance to Earth of 8.4 kpc, this allows an accurate estimation of the incident luminosity for both the *Chandra* data from 2000 and the *XMM-Newton data* from 2004. The *Chandra* data giving an incident luminosity of $L_{2-10} = 1.57 \times 10^{39}$ erg/s and the *XMM-Newton* data giving an incident luminosity $L_{2-10} = 1.13 \times 10^{39}$ erg/s (see tables 5.6 & 5.8).

As mentioned in Chapter 5, the Gaussian fit parameters were obtained from Protheroe et al. (2008). They were a cloud radius of 12 pc and a standard deviation $\sigma = 2.75$ pc, with a step size $\kappa = 1.2$ pc. The fit quality is still borderline acceptable, but not as good as the uniform density fit, having a $\chi^2_{red} = 1.33\,(618)$. The fitted hydrogen column density is found to be $N_H = 2.5^{+0.43}_{-0.66} \times 10^{24}$ cm^{-2} and the photon index is found to be $\Gamma = 1.83^{+0.06}_{-0.11}$. For the positioning, the fitted angular position $\theta = 89^{+10}_{-11}$ degrees implies that Sgr B2 lies approximately at its projected distance from Sgr $A^\star$ of 100^{+2}_{-2} pc. For the luminosity, the *Chandra* data gives $L_{2-10} = 2.03 \times 10^{39}$ erg/s. The *XMM-Newton* data gives $L_{2-10} = 1.48 \times 10^{39}$ erg/s (see tables 5.7 & 5.9). A point of particular note is that in both fits the hard X-ray data is fundamental in properly constraining the spectral index. Conversely, it does increase the chi-square value of the fits.

6.2 Literature Comparison

There have been several attempts to constrain the parameters of Sgr B2 undertaken in the literature previously. They were discussed in Chapter 1 and will be reiterated in this section, with the results obtained from this work discussed in the context of the literature. The column density predicted by both models ($N_{H_c} = 2.13 \times 10^{24}$ cm^{-2}) and ($N_{H_G} = 2.5 \times 10^{24}$ cm^{-2}) respectively, is higher than those that have been obtained with previous X-ray observations, where the value usually obtained is on the order of 10^{23} cm^{-2} (Terrier et al., 2010; Capelli et al., 2012; Zhang et al., 2015). This lower estimation is most likely due to the improper XRN models used by this earlier work. However, it is in good agreement with radio observations, for example Protheroe et al. (2008); Jones et al. (2011), who also give hydrogen column densities on the order of 10^{24} cm^{-2}. The *wabs* column density, as mentioned before, represents the column density along the line of sight between Sgr B2 and Earth. This column density is a factor of about 30 - 40 times lower than the column density of the cloud, representing less than 3% of the total column density encountered by the emission. Thus, is not thought to contribute significantly to the final observed spectrum. However, it is also in agreement with values given in the literature (Zhang et al., 2015; Mori et al., 2015). The photon index found by both models $\Gamma_C = 2.16^{+0.12}_{-0.16}$ and $\Gamma_G = 1.83^{+0.06}_{-0.11}$ (depending on the density distribution) is in good agreement with

the literature, for example; 1.9 (Revnivtsev et al., 2004), 2.0 (Terrier et al., 2010), 2.5 (Nobukawa et al., 2011), and 2.0 (Zhang et al., 2015).

The fitted angular position is the most varied between the different density distribution fits, perhaps not surprisingly given that it is the least constrained parameter in the fits. However, it does present the question of which result to trust, this issue is discussed further in Section 6.4. The fitted position for the uniform fit of $\approx 64°$ is in nominal agreement with some previous estimates, but not others. The VLBI parallax measurements made by Reid et al. (2009), placed Sgr B2 (assuming it is on a nearly circular Galactic orbit) ≈ 130 pc closer to us than Sgr A*, which equates to an angular position of $\approx 37°$. Although they are in agreement that Sgr B2 lies on the near side of Sgr A*, they cannot be said to be in agreement with the results presented in this thesis. Their measurement, however, is dependent on a precise knowledge of the exact distance to Sgr A* itself, which they take to be 8 kpc. However, this distance to Sgr A* is far from certain, with estimates ranging from 7.2 kpc (Bica et al., 2006) to 8.7 kpc (Vanhollebeke et al., 2009). Conversely, the measurements presented here have no such requirement. The orbital motion model of Kruijssen et al. (2015) gives a Sgr B2 distance of ≈ 38 parsecs closer to Earth than Sgr A*, which is in nominal agreement with the 48^{+2}_{-1} pc that the uniform density fit gives. Unfortunately, they do not give error estimations for this value, thus whether the two results are within error of each other is unknown. Ryu et al. (2009) used partial covering of the plasma emission in the CMZ to try to place various MC's along the line of sight, they also found Sgr B2 to be on the near side of Sgr A*. While they do not give exact values, from a cursory look at Figure 7 from Ryu et al. (2009), the position of Sgr B2 does appear to be in nominal agreement with that presented here. Finally, Odaka et al. (2011) also provisionally suggested Sgr B2 is located on the near side, though they did not perform data fitting.

The fitted angular position for the Gaussian distribution is $\theta = 89^{+10}_{-11}$ degrees, which places Sgr B2 in line with Sgr A*, neither on the near or far side. No similar work in the literature has placed Sgr B2 in this position. Molinari et al. (2011) used *Herschel* Infrared observations and a simple twisted ellipse model, and found Sgr B2 to be located far behind Sgr A*. However, their work was subject to a now disproved (Kruijssen et al., 2015) assumption of circular motion.

Given the close values of the N_H between the uniform and Gaussian models, it is not surprising that they give very similar incident luminosity results. The

estimated incident luminosity on the order 10^{39} erg/s is in good agreement with the literature. Terrier et al. (2010) gave an incident luminosity of $L_{1-100} = 1.5 - 5 \times 10^{39}$ erg/s, which is in good agreement with the results presented here. However, they used the Reid et al. (2009) parallax measurements for their Sgr A* - Sgr B2 distance, which might explain their maximum going 50% higher than these results, given the extra distance. They also operated under the assumption of a lower N_H, which would explain the lower L_{2-10} luminosity. Zhang et al. (2015) gave a luminosity of $L_{1-100} = 0.7 \times 10^{39}$ erg/s which is again a lower L_{2-10}, but is again explained by the lower N_H used, on the order of 10^{23} cm^{-2}.

6.3 Iron Abundance

A comparison of the Monte Carlo code output for different iron abundances was presented in Section 3.6. It is of note that the table models do not allow iron abundance as either a free or frozen parameter. With regard to the abundance of Sgr B2, many authors give an iron abundance greater than solar, for example the most recent measurement by Terrier et al. (2010) gave a value of ≈ 1.3 times solar, while earlier results from Murakami et al. (2001) & Revnivtsev et al. (2004) gave ≈ 1.9 times solar. Although the results presented in Chapter 5 did not include changing iron abundance results, several table models for fitting were created with different iron abundances. For example, a 1.9 times solar iron abundance uniform density table model was fit to the data.

Although the fit of this model is technically inferior, $\chi^2_{red} = 1.33\,(618)$ compared to $\chi^2_{red} = 1.18\,(618)$ of the solar abundance model, it does better reproduce relative flux of the iron line. This does suggest the abundance is higher than solar levels and the fitted parameters were found to be quite similar, which is not surprising given the small differences in low energy continuum, as shown by Figure 3.17. However, those small differences in low energy flux do lead to the calculated incident luminosity L_{2-10} increasing by $\approx 8\%$. After considering several factors, namely: the similar parameters, the fact that at higher iron abundances most changes in the continuum come at energies higher than 7 keV, the fact that a proper equivalent width analysis is not within the scope of this project, the uncertain nature of the iron abundance value, the worse fit, and the desire to keep the number of fitting parameters as low as possible, it was decided to leave all

abundances at solar levels. However, the question of iron abundance does remain open, especially with regard to the incident luminosity.

6.4 Density Profile Determination

The results presented thus far have not differentiated between density profiles. Figure 3.14 from Section 3.5 shows the changes in output spectra with changing density profiles for a sphere of constant mass. The differences are quite small; only the e^{-r} has a discrepancy that is noticeably significant. In fact, it is unlikely that current X-ray data is capable of determining the density profiles of MCs by this method alone. Consequently, the density profile is not a free parameter within the table models. Additionally, the low energy (1 - 10 keV) part of the spectrum is more sensitive to changes in density profile than the high energy ($>$ 10 keV), given good quality data it may be possible to use this method to determine the density profile.

With regard to Sgr B2 specifically, the actual density profile is most certainly not uniform, as shown by multiple authors (Etxaluze et al., 2013; Jones et al., 2011; Protheroe et al., 2008). As mentioned previously, the Sgr B2 data was fit with a Gaussian density table model based on the parameters found by Protheroe et al. (2008). Both the uniform density and the Gaussian density fit the data reasonably well. Importantly, they predict similar parameters. However, the uniform density fit is technically superior with a $\chi^2_{red} = 1.18\,(618)$ as opposed to $\chi^2_{red} = 1.33\,(618)$ for the Gaussian. This inferior Gaussian fit is perhaps due to the small centralized nature of the observations, which preclude treating the observation as that of a cloud with a Gaussian distribution, given the large size of the cloud. The observations used are primarily of the dense central core of Sgr B2, whilst the Gaussian Model was for a far larger cloud. Indeed, it is possible that the dense core of Sgr B2 is so compact (R $\approx$ 2 pc), that the uniform density model is more than adequately reproducing its output spectrum. However, as a result of a combination of several points: the knowledge that Sgr B2 likely has a Gaussian density distribution, the small observation size, the fact that the uniform density fit is a superior fit, and that the fitted parameters are quite similar, it was decided not to try to distinguish between the results, either in Walls et al. (2016) or here.

6.5 Impact of Time Dependance

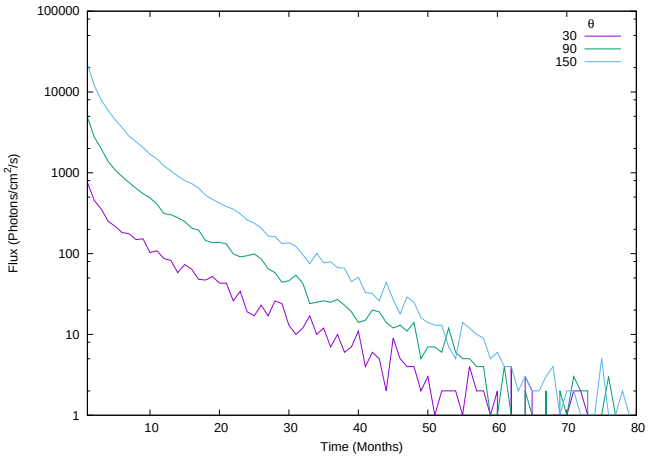
As discussed in Section 3.7 there remains a significant open question around the time dependence effects. The model presented here does not include timing information. All the photons are treated as entering the cloud at the same time and leaving it at the same time. Given how how large a divergence this is from reality, the question is far from insignificant. The effects in general are shown in Section 3.7.

However, the discussion presented here is with regard to the Sgr B2 fitting in particular. Primarily, more recent observations (such as *NuSTAR*) of Sgr B2 could not be used directly with this model since it assumes the cloud is at, or close to, full illumination, that is to say the cloud is currently being illuminated by the incident flare. The hard X-ray flux has decreased by 40% from 2003 to 2010 as shown by Terrier et al. (2010), with a time constant consistent with the light crossing time of the cloud. This linear decrease in flux is, given the cloud optical thickness, mostly likely due to multiply scattered photons that are still within the cloud. Sunyaev and Churazov (1998) showed how multiple scattering can cause photons to be observed long after the incident flare has passed through the cloud, on the order of the light crossing time of the cloud. From the 2003 - 2010 observations by Terrier et al. (2010), it is clear that the main illuminating front has already started to leave the cloud. The relative brightness of the densest cores observed by *NuSTAR* in hard X-rays (Zhang et al., 2015) also supports this.

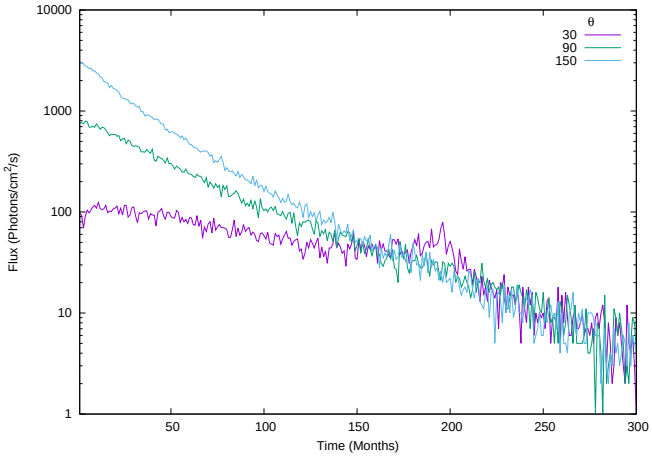
Therefore, modelling the output spectra at later times would require properly taking into account the time dependent effects. However, for the time interval between the observations used in this project, namely 2000 - 2004, there is not expected to be any significant change in spectral shape. Figure 6.1 shows the light curves of observed photons in the 1-5 keV, 5-15 keV and the 15-50 keV energy ranges for a Compton thick ($N_H = 4 \times 10^{24}$ cm^{-2}) cloud for various line of sight positions. Importantly, in Figure 6.1 a), the low energy light curve, there is an extremely rapid decay in soft X-ray flux after the cessation of the incident flare. In fact, this decay rate is best described by a power law, particularly in the first 20 months. Given the low characteristic time scale in the low energy, the observed time behaviour is likely close to that of the illuminating source. As there is an order of magnitude decay in the first year after flare cessation and there is a rather marginal flux variation observed in 2004 compared to 2000, it

can be inferred that the illuminating flare lasted at least 4 years or close to it. In this case the spectral shape will not change between observations, assuming the incident spectrum remains the same.

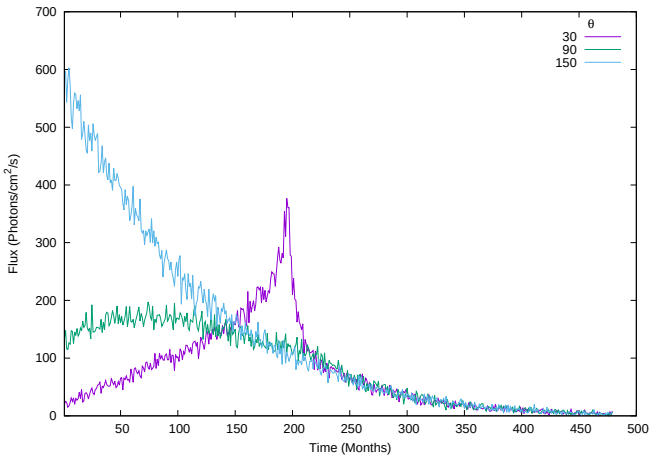
An interesting observation in the higher energy light curves, especially the 15-50 keV case, is the rapid and large flux increase in the low angle case around 200 months, that quickly falls off to join the other angular cases, which are simply emitting multiply scattered photons. These flux increases are coincident exactly with the light crossing time of the cloud and represent the light front leaving the cloud. As this is only visible in the low angle case, and would surely be detectable, it could be a useful sign in determining if a cloud has a low angular position on the line of sight.



(a)



(b)



(c)

Figure 6.1: Light curves of output flux from a Compton thick uniform density cloud, for various cloud angular positions. a) 1 - 5 keV, b) 5 - 15 keV, c) 15 - 50 keV.

Chapter 7

Conclusions and Future Prospects

The study of the Galactic centre is an ongoing and important pursuit in the field of modern astrophysics, especially in the domain of X-rays, which have only become a useful scientific tool in the past few decades. In particular, the study of the supermassive black hole at the centre of the Galaxy, known as Sgr A*, is an exciting area, especially because Sgr A* is an unusual supermassive black hole. It is quite dim, and probing its far and recent past is an area of ongoing study. One of the most investigated questions currently is, whether Sgr A* was a far more active emitter in the recent past. The main reason for believing it was more active, comes in the form of reflected X-rays from the giant molecular clouds in the Galactic centre region. However, a main point of uncertainty in analysing this reflected emission is that the three dimensional positions of these giant molecular clouds remains largely unknown.

In this PhD thesis I have attempted to solve this problem via X-ray reflection simulations and I have presented a new Monte Carlo code for simulating X-ray reflection spectra from molecular clouds. The code is capable of modelling clouds of varying density distributions and of varying geometries. I have shown how different input parameters will result in highly divergent output spectra and provide an analysis of the processes behind these changes. Using *Chandra*, *XMM-Newton* and *INTEGRAL* observations, several parameters of the giant molecular cloud Sgr B2 were constrained using XSPEC table models. These are the photon index Γ of the incident spectrum, the luminosity of the illuminating source, the N_H of the reflecting cloud and most importantly the angular position of the cloud relative to the line of sight. The results of this fitting are summarised succinctly by Figure 7.1 where the best estimated location of Sgr B2 relative to Sgr A* is

shown. There remains a question of uncertainty around the density distribution of this cloud and for this reason the positions determined by fitting the cloud with both a uniform and a Gaussian density distribution are shown in Figure 7.1.

These results are in good agreement with similar results from the literature, and provide the best estimates to date of the Sgr B2 line of sight position and, by extension, the luminosity of the incident flare from Sgr A*. The ultimate aim of this work is to facilitate the further analysis of GMCs in the Galactic centre region as reflectors of previous outbursts or periods of higher activity from Sgr A*, primarily by knowing the line of sight angular positions of the clouds. The XSPEC table models created by the work presented in this thesis are thus made freely available to all on the CERN scientific repository Zenodo, available at the following url http://dx.doi.org/10.5281/zenodo.60229.

What about the future of this line of research? The use of this code is anticipated in further work, fitting to other broadband X-ray data (such as NuSTAR) from various giant molecular clouds in the Galactic centre. Helping to determine their characteristics, and creating a three dimensional picture of the Galactic centre, thus offering improved constraints on the time delay and duration of the illuminating events.

In terms of improving the code, there are two primary areas that certainly require further work and investigation. The question of the time dependence is an important one; the code makes certain approximations regarding the time dependence that diverge from reality and are a possible source of error in the fittings performed with the code. It is planned that future versions of the code will be updated to account for this time dependence question. Once the code is properly taking these time dependence effects into account, it will be possible to use it with more recent data of Sgr B2 from high resolution instruments such as *NuSTAR*. This high quality data will hopefully allow for a determination of the GMC density distributions and thus provide even better constraints on the line of sight positions.

The second area, is that of polarisation. The scattering by electrons, as simulated by the code presented in this thesis, will induce a linear polarisation on the reflected emission. In fact, there is a strong angular dependence on the degree of polarisation, which is given by $doP = (1 - \cos(2\theta))/(1 + \cos(2\theta))$, where θ is the angle of scatter (Basko et al., 1974). The polarised light from the molecular clouds in the Galactic centre will allow an accurate determination of the cloud's

line of sight position as shown by (Odaka et al., 2011). When this project originally started it was planned to include the effect of polarisation in the output, however time sometimes conspires against us and this was not feasible. Combined with the unfortunate demise of the Astro-H satellite, which was to provide detailed polarisation maps of the Galactic centre, the community must continue to wait for these results. That is until the launch of the next X-ray polarimetry mission, one such mission, XIPE, is currently under assessment at ESA.

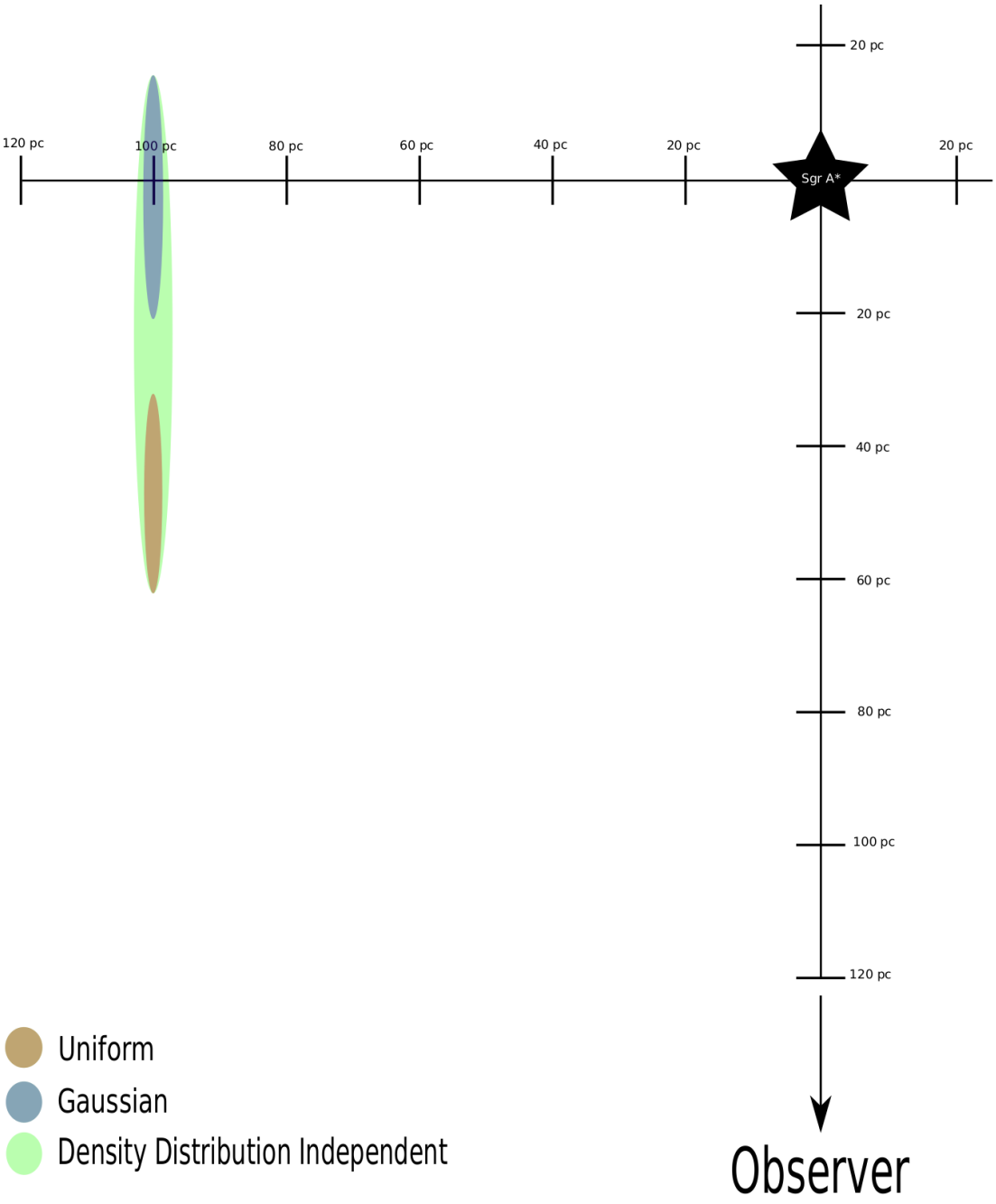


Figure 7.1: Looking down onto the Galactic plane from above. Best fit estimation for position of Sgr B2 relative to Sgr A*. Position with 90% confidence for both the Uniform and Gaussian fits shown, as well as the best estimated position regardless of density distribution.

Bibliography

M. S. Longair. *High Energy Astrophysics*. Cambridge University Press, 2011.

N. E. Kassim, T. N. Larosa, T. J. W. Lazio, and S. D. Hyman. Wide Field Radio Imaging of the Galactic Center. In *The Central Parsecs of the Galaxy*, volume 186 of *Astronomical Society of the Pacific Conference Series*, page 403, June 1999.

G. Ponti, R. Terrier, A. Goldwurm, G. Belanger, and G. Trap. Discovery of a Superluminal Fe K Echo at the Galactic Center: The Glorious Past of Sgr A* Preserved by Molecular Clouds. *ApJ*, 714:732–747, May 2010.

M. Walls, M. Chernyakova, R. Terrier, and A. Goldwurm. Examining molecular clouds in the galactic centre region using x-ray reflection spectra simulations. *MNRAS*, 463(3):2893–2903, 2016.

R. Giacconi, H. Gursky, F. R. Paolini, and B. B. Rossi. Evidence for x Rays From Sources Outside the Solar System. *Physical Review Letters*, 9:439–443, December 1962.

W. Forman, C. Jones, L. Cominsky, P. Julien, S. Murray, G. Peters, H. Tananbaum, and R. Giacconi. The fourth Uhuru catalog of X-ray sources. *ApJS*, 38: 357–412, December 1978.

W. Voges, B. Aschenbach, T. Boller, H. Bräuninger, U. Briel, W. Burkert, K. Dennerl, J. Englhauser, R. Gruber, F. Haberl, G. Hartner, G. Hasinger, M. Kürster, E. Pfeffermann, W. Pietsch, P. Predehl, C. Rosso, J. H. M. M. Schmitt, J. Trümper, and H. U. Zimmermann. The ROSAT all-sky survey bright source catalogue. *A&A*, 349:389–405, September 1999.

G. H. Rieke, M. J. Rieke, and A. E. Paul. Origin of the excitation of the galactic center. *ApJ*, 336:752–761, January 1989.

M. J. Reid, K. M. Menten, X. W. Zheng, A. Brunthaler, and Y. Xu. A Trigonometric Parallax of Sgr B2. *ApJ*, 705:1548–1553, November 2009.

R. D. Blum, D. L. Depoy, and K. Sellgren. A comparison of near-infrared spectra of the galactic center compact He I emission-line sources and early-type mass-losing stars. *ApJ*, 441:603–616, March 1995.

M. Morris and E. Serabyn. The Galactic Center Environment. *ARA&A*, 34: 645–702, 1996.

G. Dahmen, S. Huttemeister, T. L. Wilson, and R. Mauersberger. Molecular gas in the Galactic center region. II. Gas mass and $N_H/I_{12_{CO}}$ conversion based on a $C^{18}O(\mathrm{J} = 1 \text{-}\!\!\rightarrow 0)$ survey. *A&A*, 331:959–976, March 1998.

R. Genzel, N. Thatte, A. Krabbe, H. Kroker, and L. E. Tacconi-Garman. The Dark Mass Concentration in the Central Parsec of the Milky Way. *ApJ*, 472: 153, November 1996.

R. Genzel, D. Hollenbach, and C. H. Townes. The nucleus of our Galaxy. *Reports on Progress in Physics*, 57:417–479, May 1994.

Z.-Q. Shen, K. Y. Lo, M.-C. Liang, P. T. P. Ho, and J.-H. Zhao. A size of ˜1AU for the radio source Sgr A* at the centre of the Milky Way. *Nature*, 438:62–64, November 2005.

R. Schödel, T. Ott, R. Genzel, R. Hofmann, M. Lehnert, A. Eckart, N. Mouawad, T. Alexander, M. J. Reid, R. Lenzen, M. Hartung, F. Lacombe, D. Rouan, E. Gendron, G. Rousset, A.-M. Lagrange, W. Brandner, N. Ageorges, C. Lidman, A. F. M. Moorwood, J. Spyromilio, N. Hubin, and K. M. Menten. A star in a 15.2-year orbit around the supermassive black hole at the centre of the Milky Way. *Nature*, 419:694–696, October 2002.

R. Schödel, D. Merritt, and A. Eckart. The nuclear star cluster of the Milky Way: proper motions and mass. *A&A*, 502:91–111, July 2009.

S. Gillessen, F. Eisenhauer, S. Trippe, T. Alexander, R. Genzel, F. Martins, and T. Ott. Monitoring Stellar Orbits Around the Massive Black Hole in the Galactic Center. *ApJ*, 692:1075–1109, February 2009.

R. Genzel, F. Eisenhauer, and S. Gillessen. The Galactic Center massive black hole and nuclear star cluster. *Reviews of Modern Physics*, 82:3121–3195, October 2010.

G. Ponti, M. R. Morris, R. Terrier, and A. Goldwurm. Traces of Past Activity in the Galactic Centre. In *Cosmic Rays in Star-Forming Environments*, volume 34 of *Astrophysics and Space Science Proceedings*, page 331, 2013.

M. N. Pavlinsky, S. A. Grebenev, and R. A. Sunyaev. X-ray images of the Galactic Center obtained with ART-P/GRAN AT: Discovery of new sources, variability of persistent sources, and localization of X-ray bursters. *ApJ*, 425:110–121, April 1994.

A. Goldwurm, B. Cordier, J. Paul, J. Ballet, L. Bouchet, J.-P. Reques, G. Vedrenne, P. Mandrou, R. Sunyaev, E. Churazov, M. Gilfanov, A. Finogenov, A. Vikhlinin, A. Dyachkov, N. Khavenson, and V. Kovtunenko. Possible evidence against a massive black hole at the Galactic Centre. *Nature*, 371:589–591, October 1994.

N. Degenaar, R. Wijnands, E. M. Cackett, J. Homan, J. J. M. in't Zand, E. Kuulkers, T. J. Maccarone, and M. van der Klis. A four-year XMM-Newton/Chandra monitoring campaign of the Galactic centre: analysing the X-ray transients. *A&A*, 545:A49, September 2012.

F. K. Baganoff, Y. Maeda, M. Morris, M. W. Bautz, W. N. Brandt, W. Cui, J. P. Doty, E. D. Feigelson, G. P. Garmire, S. H. Pravdo, G. R. Ricker, and L. K. Townsley. Chandra X-Ray Spectroscopic Imaging of Sagittarius A* and the Central Parsec of the Galaxy. *ApJ*, 591:891–915, July 2003.

K. Koyama, H. Awaki, H. Kunieda, S. Takano, and Y. Tawara. Intense 6.7-keV iron line emission from the Galactic Centre. *Nature*, 339:603–605, June 1989.

M. Revnivtsev. Distribution of the Galactic bulge emission at $|b| > 2^\circ$ according to the RXTE Galactic Center scans. *A&A*, 410:865–870, November 2003.

K. Koyama, T. Inui, Y. Hyodo, H. Matsumoto, T. G. Tsuru, Y. Maeda, H. Murakami, S. Yamauchi, S. E. Kissel, K.-W. Chan, and Y. Soong. Discoveries of Diffuse Iron Line Sources from the Sgr B Region. *PASJ*, 59:221–227, January 2007.

R. A. Sunyaev, M. Markevitch, and M. Pavlinsky. The center of the Galaxy in the recent past - A view from GRANAT. *ApJ*, 407:606–610, April 1993.

K. Koyama, Y. Maeda, T. Sonobe, T. Takeshima, Y. Tanaka, and S. Yamauchi. ASCA View of Our Galactic Center: Remains of Past Activities in X-Rays? *PASJ*, 48:249–255, April 1996.

R. L. Brown and R. J. Gould. Interstellar Absorption of Cosmic X Rays. *Phys. Rev. D*, 1:2252–2256, April 1970.

M. M. Basko, R. A. Sunyaev, and L. G. Titarchuk. Reflection and reprocessing of X-ray source radiation by the atmosphere of the normal star in a binary system. *A&A*, 31:249–263, March 1974.

K. Nandra and I. M. George. X-Ray Reprocessing by Cold Clouds in Active Galactic Nuclei. *MNRAS*, 267:974, April 1994.

R. A. Sunyaev and E. M. Churazov. Scattering of X-ray emission lines by neutral and molecular hydrogen in the solar atmosphere and in the vicinity of active galactic nuclei and compact X-ray sources. *Astronomy Letters*, 22:648–663, September 1996.

F. K. Baganoff, M. W. Bautz, W. N. Brandt, G. Chartas, E. D. Feigelson, G. P. Garmire, Y. Maeda, M. Morris, G. R. Ricker, L. K. Townsley, and F. Walter. Rapid X-ray flaring from the direction of the supermassive black hole at the Galactic Centre. *Nature*, 413:45–48, September 2001.

A. Goldwurm, E. Brion, P. Goldoni, P. Ferrando, F. Daigne, A. Decourchelle, R. S. Warwick, and P. Predehl. A New X-Ray Flare from the Galactic Nucleus Detected with the XMM-Newton Photon Imaging Cameras. *ApJ*, 584:751–757, February 2003.

D. Porquet, P. Predehl, B. Aschenbach, N. Grosso, A. Goldwurm, P. Goldoni, R. S. Warwick, and A. Decourchelle. XMM-Newton observation of the brightest X-ray flare detected so far from Sgr A*. *A&A*, 407:L17–L20, August 2003.

G. Rockefeller, C. L. Fryer, F. Melia, and M. S. Warren. Diffuse X-Rays from the Inner 3 Parsecs of the Galaxy. *ApJ*, 604:662–670, April 2004.

H. Murakami, K. Koyama, M. Sakano, M. Tsujimoto, and Y. Maeda. ASCA Observations of the Sagittarius B2 Cloud: An X-Ray Reflection Nebula. *ApJ*, 534:283–290, May 2000.

H. Murakami, K. Koyama, and Y. Maeda. Chandra Observations of Diffuse X-Rays from the Sagittarius B2 Cloud. *ApJ*, 558:687–692, September 2001.

M. G. Revnivtsev, E. M. Churazov, S. Y. Sazonov, R. A. Sunyaev, A. A. Lutovinov, M. R. Gilfanov, A. A. Vikhlinin, P. E. Shtykovsky, and M. N. Pavlinsky. Hard X-ray view of the past activity of Sgr A* in a natural Compton mirror. *A&A*, 425:L49–L52, October 2004.

M. P. Muno, F. K. Baganoff, W. N. Brandt, S. Park, and M. R. Morris. Discovery of Variable Iron Fluorescence from Reflection Nebulae in the Galactic Center. *ApJ*, 656:L69–L72, February 2007.

V. Dogiel, K.-S. Cheng, D. Chernyshov, A. Bamba, A. Ichimura, H. Inoue, C.-M. Ko, M. Kokubun, Y. Maeda, K. Mitsuda, and N. Y. Yamasaki. Origin of 6.4keV Line Emission from Molecular Clouds in the Galactic Center. *PASJ*, 61:901–907, August 2009.

K. Koyama, T. Inui, H. Matsumoto, and T. G. Tsuru. A Time-Variable X-Ray Echo: Indications of a Past Flare of the Galactic-Center Black Hole. *PASJ*, 60: S201–S206, January 2008.

T. Inui, K. Koyama, H. Matsumoto, and T. G. Tsuru. Time Variability of the Neutral Iron Lines from the Sagittarius B2 Region and Its Implication of a Past Outburst of Sagittarius A. *PASJ*, 61:S241–S253, January 2009.

R. Terrier, G. Ponti, G. Bélanger, A. Decourchelle, V. Tatischeff, A. Goldwurm, G. Trap, M. R. Morris, and R. Warwick. Fading Hard X-ray Emission from the Galactic Center Molecular Cloud Sgr B2. *ApJ*, 719:143–150, August 2010.

M. Nobukawa, S. G. Ryu, T. G. Tsuru, and K. Koyama. New Evidence for High Activity of the Supermassive Black Hole in our Galaxy. *ApJ*, 739:L52, October 2011.

H. Nakajima, T. G. Tsuru, M. Nobukawa, H. Matsumoto, K. Koyama, H. Murakami, A. Senda, and S. Yamauchi. X-Ray Reflection Nebulae with Large

Equivalent Widths of the Neutral Iron Kα Line in the Sagittarius C Region. *PASJ*, 61:S233–S240, January 2009.

C. K. Cramphorn and R. A. Sunyaev. Interstellar gas in the Galaxy and the X-ray luminosity of Sgr A* in the recent past. *A&A*, 389:252–270, July 2002.

M. Su, T. R. Slatyer, and D. P. Finkbeiner. Giant Gamma-ray Bubbles from Fermi-LAT: Active Galactic Nucleus Activity or Bipolar Galactic Wind? *ApJ*, 724:1044–1082, December 2010.

R. Capelli, R. S. Warwick, N. Cappelluti, S. Gillessen, P. Predehl, D. Porquet, and S. Czesla. Discovery of X-ray flaring activity in the Arches cluster. *A&A*, 525:L2, January 2011.

R. Capelli, R. S. Warwick, D. Porquet, S. Gillessen, and P. Predehl. The X-ray lightcurve of Sagittarius A* over the past 150 years inferred from Fe-Kα line reverberation in Galactic centre molecular clouds. *A&A*, 545:A35, September 2012.

V. Tatischeff, A. Decourchelle, and G. Maurin. Nonthermal X-rays from low-energy cosmic rays: application to the 6.4 keV line emission from the Arches cluster region. *A&A*, 546:A88, October 2012.

Q. D. Wang, H. Dong, and C. Lang. The interplay between star formation and the nuclear environment of our Galaxy: deep X-ray observations of the Galactic centre Arches and Quintuplet clusters. *MNRAS*, 371:38–54, September 2006.

M. Clavel, S. Soldi, R. Terrier, V. Tatischeff, G. Maurin, G. Ponti, A. Goldwurm, and A. Decourchelle. Variation of the X-ray non-thermal emission in the Arches cloud. *MNRAS*, 443:L129–L133, September 2014a.

G. Rockefeller, C. L. Fryer, F. K. Baganoff, and F. Melia. The X-Ray Ridge Surrounding Sagittarius A* at the Galactic Center. *ApJ*, 635:L141–L144, December 2005.

C. L. Fryer, G. Rockefeller, A. Hungerford, and F. Melia. The Sgr B2 X-Ray Echo of the Galactic Center Supernova Explosion that Produced Sgr A East. *ApJ*, 638:786–796, February 2006.

M. Clavel, R. Terrier, A. Goldwurm, M. R. Morris, G. Ponti, S. Soldi, and G. Trap. The reflection of two past outbursts of Sagittarius A* observed by Chandra during the last decade. In *The Galactic Center: Feeding and Feedback in a Normal Galactic Nucleus*, volume 303 of *IAU Symposium*, pages 344–348, May 2014b.

G. Witzel, A. Eckart, M. Bremer, M. Zamaninasab, B. Shahzamanian, M. Valencia-S., R. Schödel, V. Karas, R. Lenzen, N. Marchili, N. Sabha, M. Garcia-Marin, R. M. Buchholz, D. Kunneriath, and C. Straubmeier. Source-intrinsic near-infrared properties of sgr a*: Total intensity measurements. *The Astrophysical Journal Supplement Series*, 203(2):18, 2012.

J. Cuadra, S. Nayakshin, and F. Martins. Variable accretion and emission from the stellar winds in the Galactic Centre. *MNRAS*, 383:458–466, January 2008.

Y.-W. Yu, K. S. Cheng, D. O. Chernyshov, and V. A. Dogiel. A past capture event at Sagittarius A* inferred from the fluorescent X-ray emission of Sagittarius B clouds. *MNRAS*, 411:2002–2008, March 2011.

K. Zubovas and S. Nayakshin. Fermi bubbles in the Milky Way: the closest AGN feedback laboratory courtesy of Sgr A*? *MNRAS*, 424:666–683, July 2012.

S. Sazonov, R. Sunyaev, and M. Revnivtsev. Coronal radiation of a cusp of spun-up stars and the X-ray luminosity of Sgr A*. *MNRAS*, 420:388–404, February 2012.

J.-H. Zhao, W. M. Goss, and P. T. P. Ho. VLA OH Observations of High Negative Velocity Gas toward Sagittarius A West: A High-Velocity Cloud Interacting with the Galactic Center. *ApJ*, 450:122, September 1995.

A. M. Ghez, S. D. Hornstein, J. R. Lu, A. Bouchez, D. Le Mignant, M. A. van Dam, P. Wizinowich, K. Matthews, M. Morris, E. E. Becklin, R. D. Campbell, J. C. Y. Chin, S. K. Hartman, E. M. Johansson, R. E. Lafon, P. J. Stomski, and D. M. Summers. The First Laser Guide Star Adaptive Optics Observations of the Galactic Center: Sgr A*'s Infrared Color and the Extended Red Emission in its Vicinity. *ApJ*, 635:1087–1094, December 2005.

S. G. Ryu, K. Koyama, M. Nobukawa, R. Fukuoka, and T. G. Tsuru. An X-Ray Face-On View of the Sagittarius B Molecular Clouds Observed with Suzaku. *PASJ*, 61:751–761, August 2009.

T. Sawada, T. Hasegawa, T. Handa, and R. J. Cohen. A molecular face-on view of the Galactic Centre region. *MNRAS*, 349:1167–1178, April 2004.

R. Sunyaev and E. Churazov. Equivalent width, shape and proper motion of the iron fluorescent line emission from molecular clouds as an indicator of the illuminating source X-ray flux history. *MNRAS*, 297:1279–1291, July 1998.

S. Zhang, C. J. Hailey, K. Mori, M. Clavel, R. Terrier, G. Ponti, A. Goldwurm, F. E. Bauer, S. E. Boggs, F. E. Christensen, W. W. Craig, F. A. Harrison, J. Hong, M. Nynka, S. Soldi, D. Stern, J. A. Tomsick, and W. W. Zhang. Hard X-Ray Morphological and Spectral Studies of the Galactic Center Molecular Cloud Sgr B2: Constraining Past Sgr A* Flaring Activity. *ApJ*, 815:132, December 2015.

K. Mori, C. J. Hailey, R. Krivonos, J. Hong, G. Ponti, F. Bauer, K. Perez, M. Nynka, S. Zhang, J. A. Tomsick, D. M. Alexander, F. K. Baganoff, D. Barret, N. Barrière, S. E. Boggs, A. M. Canipe, F. E. Christensen, W. W. Craig, K. Forster, P. Giommi, B. W. Grefenstette, J. E. Grindlay, F. A. Harrison, A. Hornstrup, T. Kitaguchi, J. E. Koglin, V. Luu, K. K. Madsen, P. H. Mao, H. Miyasaka, M. Perri, M. J. Pivovaroff, S. Puccetti, V. Rana, D. Stern, N. J. Westergaard, W. W. Zhang, and A. Zoglauer. NuSTAR Hard X-Ray Survey of the Galactic Center Region I: Hard X-Ray Morphology and Spectroscopy of the Diffuse Emission. *ApJ*, 814:94, December 2015.

R. J. Protheroe, J. Ott, R. D. Ekers, D. I. Jones, and R. M. Crocker. Interpretation of radio continuum and molecular line observations of Sgr B2: free-free and synchrotron emission, and implications for cosmic rays. *MNRAS*, 390:683–692, October 2008.

S. Molinari, J. Bally, A. Noriega-Crespo, M. Compiègne, J. P. Bernard, D. Paradis, P. Martin, L. Testi, and M. Barlow. A 100 pc Elliptical and Twisted Ring of Cold and Dense Molecular Clouds Revealed by Herschel Around the Galactic Center. *ApJ*, 735:L33, July 2011.

J. M. D. Kruijssen, J. E. Dale, and S. N. Longmore. The dynamical evolution of molecular clouds near the Galactic Centre - I. Orbital structure and evolutionary timeline. *MNRAS*, 447:1059–1079, February 2015.

Hirokazu Odaka, Felix Aharonian, Shin Watanabe, Yasuyuki Tanaka, Dmitry Khangulyan, and Tadayuki Takahashi. X-ray diagnostics of giant molecular clouds in the galactic center region and past activity of sgr a*. *The Astrophysical Journal*, 740(2):103, 2011.

M. Molaro, R. Khatri, and R. A. Sunyaev. Probing the clumping structure of giant molecular clouds through the spectrum, polarisation and morphology of X-ray reflection nebulae. *A&A*, 589:A88, April 2016.

R. Eckhardt. Stan Ulam, John von Neumann, and the Monte Carlo method. *Los Alamos Sci*, 15:131–137, 1987.

Nicholas Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

D. A. Leahy and J. Creighton. Montecarlo Simulations of X-Ray Spectra for Internally Illuminated Spherical Matter Distributions. *MNRAS*, 263:314, July 1993.

K. Lodders. Solar System Abundances and Condensation Temperatures of the Elements. *ApJ*, 591:1220–1247, July 2003.

W. Bambynek, B. Crasemann, R. W. Fink, H.-U. Freund, H. Mark, C. D. Swift, R. E. Price, and P. V. Rao. X-Ray Fluorescence Yields, Auger, and Coster-Kronig Transition Probabilities. *Reviews of Modern Physics*, 44:716–813, October 1972.

J. S. Kaastra and R. Mewe. X-ray emission from thin plasmas. I - Multiple Auger ionisation and fluorescence processes for Be to Zn. *A&AS*, 97:443–482, January 1993.

J. A. Bearden and A. F. Burr. Reevaluation of X-Ray Atomic Energy Levels. *Reviews of Modern Physics*, 39:125–142, January 1967.

O. Klein and T. Nishina. Über die Streuung von Strahlung durch freie Elektronen nach der neuen relativistischen Quantendynamik von Dirac. *Zeitschrift fur Physik*, 52:853–868, November 1929.

G. B. Rybicki and A. P. Lightman. *Radiative Processes in Astrophysics*. Wiley-VCH, 1985.

W. Dunn and J. K Shultis. *Exploring Monte Carlo Methods*. Elsevier Science, 2011.

M. Molaro, R. Khatri, and R. A. Sunyaev. A thin diffuse component of the Galactic ridge X-ray emission and heating of the interstellar medium contributed by the radiation of Galactic X-ray binaries. *A&A*, 564:A107, April 2014.

J. Hubbell. Summary of existing information on the incoherent scattering of photons, particularly on the validity of the use of the incoherent scattering function. *Radiation Physics and Chemistry*, 50(1):113 – 124, 1997.

J. H. Hubbell, W. J. Veigele, E. A. Briggs, R. T. Brown, D. T. Cromer, and R. J. Howerton. Atomic form factors, incoherent scattering functions, and photon scattering cross sections. *Journal of Physical and Chemical Reference Data*, 4: 471–538, July 1975.

Tom Schoonjans, Antonio Brunetti, Bruno Golosio, Manuel Sanchez del Rio, Vicente Armando Solé, Claudio Ferrero, and Laszlo Vincze. The xraylib library for x-ray–matter interactions. recent developments. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 66(11–12):776 – 784, 2011. ISSN 0584-8547.

R. A. Sunyaev, D. B. Uskov, and E. M. Churazov. Scattering of x-ray emission lines by molecular hydrogen. *Astronomy Letters*, 25:199–205, April 1999.

A. Bielajew. *Fundamentals of the Monte Carlo method for nuetral and charge particle transport*. 2000.

K. A. Arnaud. XSPEC: The First Ten Years. In G. H. Jacoby and J. Barnes, editors, *Astronomical Data Analysis Software and Systems V*, volume 101 of *Astronomical Society of the Pacific Conference Series*, page 17, 1996.

P. R.. Bevington. *Data Reduction and Error Analysis for the Physical Sciences*. McGraw-Hill, 1969.

Y. Avni. Energy spectra of X-ray clusters of galaxies. *ApJ*, 210:642–646, December 1976.

S. L. Snowden, R. F. Mushotzky, K. D. Kuntz, and D. S. Davis. A catalog of galaxy clusters observed by XMM-Newton. *A&A*, 478:615–658, February 2008.

M. P. Muno, F. K. Baganoff, M. W. Bautz, E. D. Feigelson, G. P. Garmire, M. R. Morris, S. Park, G. R. Ricker, and L. K. Townsley. Diffuse X-Ray Emission in a Deep Chandra Image of the Galactic Center. *ApJ*, 613:326–342, September 2004.

R. K. Smith, N. S. Brickhouse, D. A. Liedahl, and J. C. Raymond. Collisional Plasma Models with APEC/APED: Emission-Line Diagnostics of Hydrogen-like and Helium-like Ions. *ApJ*, 556:L91–L95, August 2001.

Y. Tanaka, K. Koyama, Y. Maeda, and T. Sonobe. Unusual Properties of X-Ray Emission near the Galactic Center. *PASJ*, 52:L25–L30, August 2000.

M. Nobukawa, T. G. Tsuru, Y. Takikawa, Y. Hyodo, T. Inui, H. Nakajima, H. Matsumoto, K. Koyama, H. Murakami, and S. Yamauchi. Suzaku Spectroscopy of an X-Ray Reflection Nebula and a New Supernova Remnant Candidate in the SgrB1 Region. *PASJ*, 60:S191–S200, January 2008.

H. Mori, T. G. Tsuru, Y. Hyodo, K. Koyama, and A. Senda. Suzaku Observation of G359.79-0.26, a New Supernova Remnant Candidate in the Galactic Center. *PASJ*, 60:S183–S190, January 2008.

K. Wood, J. E. Bjorkman, B. A. Whitney, and A. D. Code. The Effect of Multiple Scattering on the Polarization from Axisymmetric Circumstellar Envelopes. I. Pure Thomson Scattering Envelopes. *ApJ*, 461:828, April 1996.

P. A. Jones, M. G. Burton, N. F. H. Tothill, and M. R. Cunningham. Spectral imaging of the Sagittarius B2 region in multiple 7-mm molecular lines. *MNRAS*, 411:2293–2310, March 2011.

E. Bica, C. Bonatto, B. Barbuy, and S. Ortolani. Globular cluster system and Milky Way properties revisited. *A&A*, 450:105–115, April 2006.

E. Vanhollebeke, M. A. T. Groenewegen, and L. Girardi. Stellar populations in the Galactic bulge. Modelling the Galactic bulge with TRILEGAL. *A&A*, 498: 95–107, April 2009.

M. Etxaluze, J. R. Goicoechea, J. Cernicharo, E. T. Polehampton, A. Noriega-Crespo, S. Molinari, B. M. Swinyard, R. Wu, and J. Bally. Herschel observations of the Sagittarius B2 cores: Hydrides, warm CO, and cold dust. *A&A*, 556:A137, August 2013.

Appendix A

Contour Plots

A.1 Uniform Density Fit

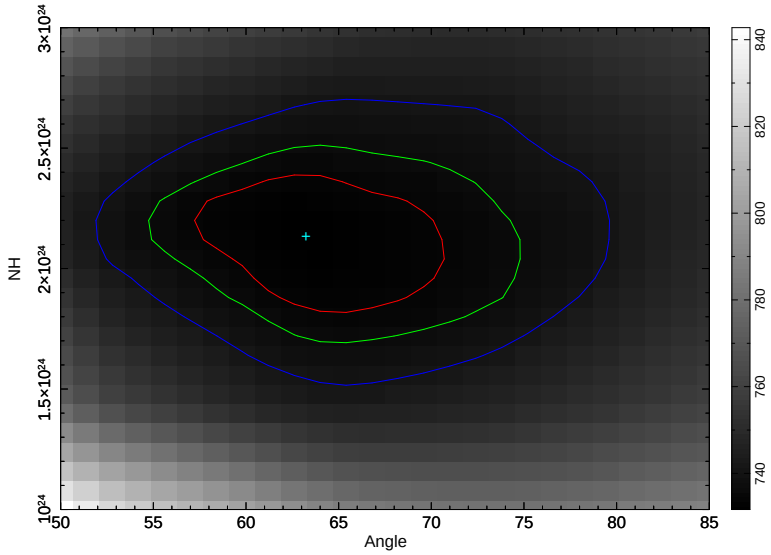


Figure A.1: Contour plot of line of sight angle θ -vs- column density N_H.

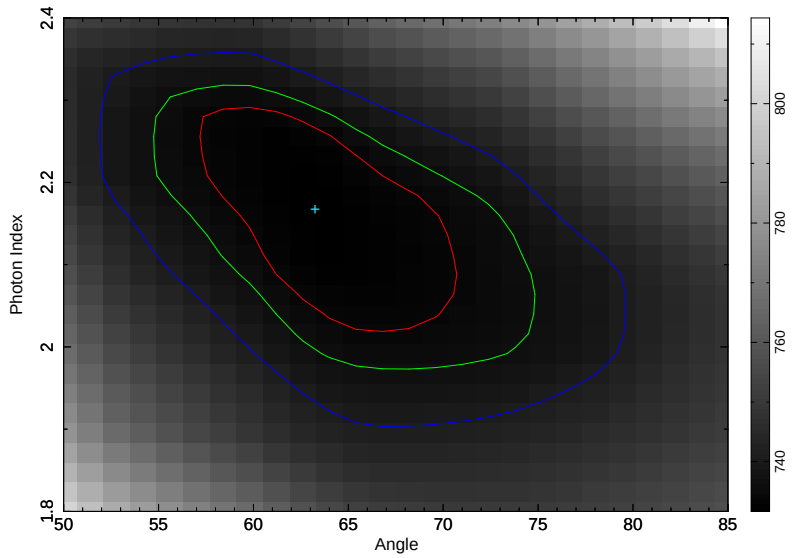


Figure A.2: Contour plot of line of sight angle θ -vs- photon index Γ

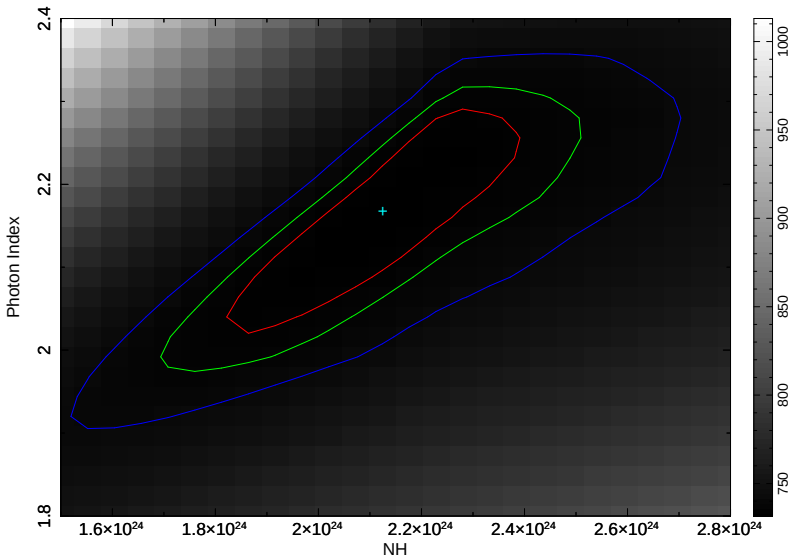


Figure A.3: Contour plot of column density N_H -vs- photon index Γ

A.2 Gaussian Density Fit

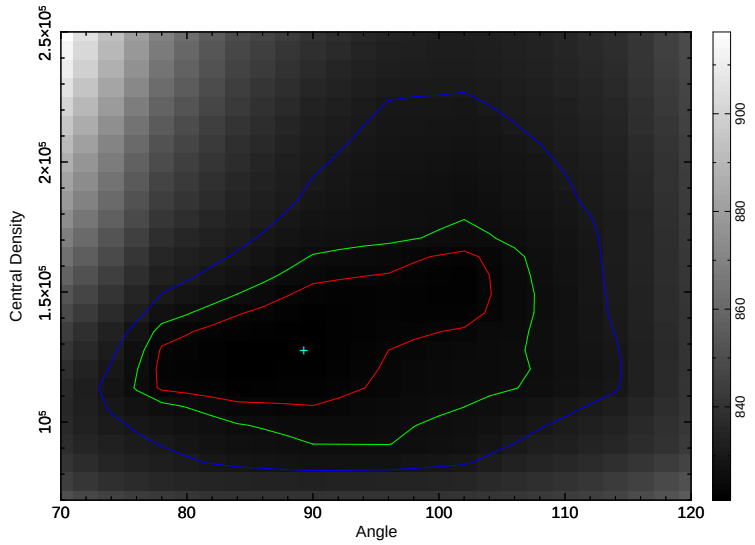


Figure A.4: Contour plot of line of sight angle θ -vs- column density N_H.

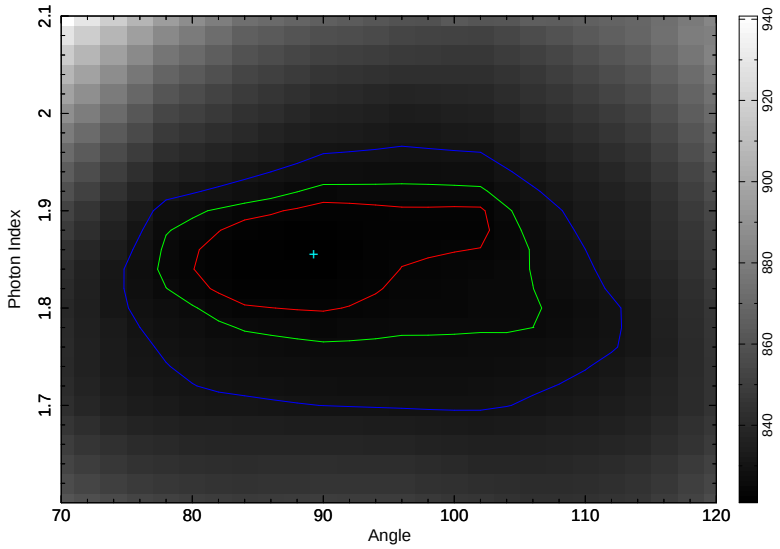


Figure A.5: Contour plot of line of sight angle θ -vs- photon index Γ

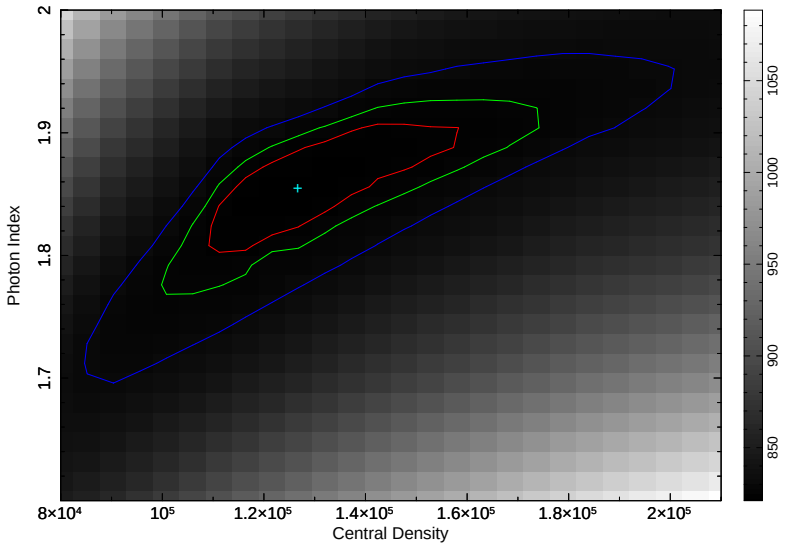


Figure A.6: Contour plot of column density N_H -vs- photon index Γ

Appendix B

Code

B.1 Main Code

B.1.1 Uniform Density Code

Snippet B.1 shows the Monte Carlo code in its entirety, the uniform density version. See Chapter 2 for summary of operation.

```
#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <string>
#include <fstream>
#include <ctime>
#include <iostream>
#include "xraylib.h"
using namespace std;

//Constants
//Physical Constants
const double myPI = atan(1) * 4.0;
const double twoPI = atan(1) * 8.0;
const double Plankconst = 6.62606957E-34;
const double Massofelectron = 9.10938291E-31;//kg
const double speedoflight = 299792458; //m/s
const double speedoflightkn = 2.99792458e10; //cm/s
const double parsec = 3.08567758e18; //cm
const double ironkbinding = 7.114; //kev
const double fyield = 0.34; // Iron kalpha fluoresecense yield
const double electronradius = 2.8179403267e-13; // cm
const double electronradiuskn = 2.8179403267e-15; // m
const double kae = 6.399; // K alpha emission energy kev
const double kbe = 7.05; //K beta emission energy kev
const double kabratio = 0.13; // Ratio of k-alpha to k-beta emission
const double barn = 1E-24;   // cm^2
const double kev2joules = 1.602176E-16;

//Code Specific Constants
const double scatlimit = 35;    //Hard limit to number of successive scatters
const bool multiplescattering = true; //Are photons aloud to multiply scatter up to scatter
    limit
const int ebins = 400; //Number of energy bins
const float ebinsf = 400.0;
const double emin = 1.0; //keV
```

```
const double emax = 400.0; //keV
const double binmin = 1.0; //keV
const double binmax = 400.0; //keV
const int abins = 360; //Number of angular bins
const double azimin = 0.0; //Theta bin min
const double azimax = twoPI; //Theta bin max
const double polarmin = 0.0; //Polar bin min in cosine
const double polarmax = 2.0; //Polar bin max in cosine
const double cloudazi = 0.0; //0.610865238 //1.57079633 //2.35619449
const double cloudpolar = (myPI/2.0);
const double clouddistance = (25.0*parsec);//To centre of cloud //cm
const double observerdistance = (840.0*parsec);//cm

//Functions
double randomnum();//Return a random number 0<x<1
double photongen(double alpha);//Generates a photon from power law
int findenergybin(double photonenergy);//Returns appropriate bin of input energy
double binenergy(int bin);//Determine energy of input bin
double rayleighscatangle(double photonenergy); //Return a Rayleigh scattering angle
double comptonscatangle(double photonenergy); //Return a Compton scattering angle
double rayleighcs(double pe); //Rayleigh scattering cross section
double comptoncs(double pe); //Compton scattering cross section
double indycs(double pe, int pos); //Photoelectric absorption cross section for an element
double totalcs(double pe); //Total photoelectric absorption cross section
double spheredistanceoutside(double px, double py, double pz, double cx, double cy, double cz
    , double azi, double polar, double cloudsize); //Ray trace sphere
double spheredistanceinside(double px, double py, double pz, double cx, double cy, double cz,
     double azi, double polar, double cloudsize); //Ray trace sphere
double distancebetweentwopoints(double x1, double y1, double z1, double x2, double y2, double
     z2);
double densityfactor(double density); //Increase density by small factor
int findazibin(double angle, double azimins[], double azimaxs[]);//Returns the angle bin
int findpolarbin(double angle, double polarmins[], double polarmaxs[]); //Returns angle bin

//Cross Section value holding arrays
double crossenergy[75];//Energies for different cross sections
double crossvalues[17][75];//Array holding various element cross sections;
double met[17];//Array holding elemental abundances
string crossfiles[19] = {"cs3/H.txt","cs3/He.txt","cs3/C.txt","cs3/N.txt","cs3/O.txt","cs3/Ne
    .txt","cs3/Na.txt","cs3/Mg.txt","cs3/Al.txt","cs3/Si.txt","cs3/S.txt","cs3/Cl.txt","cs3/
    Ar.txt","cs3/Ca.txt","cs3/Cr.txt","cs3/Fe.txt","cs3/Ni.txt","cs3/abundances.txt","cs3/
    energies_1-600_kev.txt"};

//Calculate Angle Binning levels
double azibinminlimits[abins];
double azibinmaxlimits[abins];
double polarbinminlimits[abins];
double polarbinmaxlimits[abins];

//Max arrays
double raymaxes[ebins][1];
double compmaxes[ebins][1];

//Spectra Array
float spectra[ebins][abins][abins];//Output spectrum

//Number of misses
int misstest = 0;

int main(int argc, char *argv[])
{
    //Intialise RNG
    srand(2658479);

    //Start clock
    time_t start, mid, finish;
    time(&start);

    if(argc != 7)
    {
```

```
        cout << "Wrong number of parameters, Exiting program.\nCorrect Entry is:\n1)
    Iterations, 2)Alpha, 3)Cloud Density, 4)Cloud Diamater (parsecs), 5)Iron abundance as
    factor of Solar, 6)Output file name." << endl;
        return 0;
    }

    //Command line arguemnts
    double iterations = atof(argv[1]);
    double alpha = atof(argv[2]);
    double cloudhdensity = atof(argv[3]);
    double clouddiameter = atof(argv[4])*parsec;
    double ironabundancefactor = atof(argv[5]);

    cout << "Alpha = " << alpha << endl;
    cout << "Iterations = " << iterations << endl;

    // Calculate needed values
    // Cloud column density
    double ncloud = cloudhdensity*clouddiameter;
    cout << "Cloud column density = " << ncloud << endl;
    // Cloud radius
    double cloudradius = (clouddiameter)/2.0;
    cout << "Cloud radius = " << (cloudradius/parsec) << endl;
    // Cloud angular diameter from source
    double cloudad = 2 * asin( (clouddiameter)/(2*(clouddistance)) );
    double cloudadovertwo = cloudad/2.0;
    cout << "Cloud angular diamter = " << cloudad << " Radians & " << (cloudad*(180/myPI)) <<
      " Degrees" << endl;

    //Access and bin abundances & cross sections;
    //Metallicities
    fstream infile;
    infile.open (crossfiles[17].c_str(), ios::in);
    int metplace = 0;
    for(int i = 0; i < 34; ++i)
    {
        string metinput;
        getline(infile,metinput);
        if(metinput[0] == '#')
        {
            continue;
        }
        else
        {
            met[metplace] = atof(metinput.c_str());
            ++metplace;
        }
    }
    infile.close();

    //Energy values
    infile.open (crossfiles[18].c_str(), ios::in);
    for(int i = 0; i < 75; ++i)
    {
        infile >> crossenergy[i];
    }
    infile.close();

  //Raise Iron abundance by user set amount
    cout << "Solar Fe abundance " << met[15]<< endl;
    met[15] = log10( ironabundancefactor * pow( 10.0,(met[15] - 12.0) ) ) + 12.0;
    cout << "Cloud Fe Abundance = " << met[15] << endl;

  //Elemental cross sections
    for(int i = 0; i < 17; ++i)
    {
        infile.open (crossfiles[i].c_str(), ios::in);
        for(int j = 0; j < 75; ++j)
        {
            infile >> crossvalues[i][j];
        }
```

```
    infile.close();
}

// Working constants
// Iron abundance %
double ironpercentagefactor =  1.0/pow( (10), (met[0] - met[15]));
// Speed of light by planck constant
double cbyh = speedoflight*Plankconst;
// Planck constant over mass of electron times speed of light
double homebyc = Plankconst/(Massofelectron*speedoflight);
//Factor to add contribution from non-hydrogen atoms
double denfactor = densityfactor(cloudhdensity)/cloudhdensity;
cout << "Non-Hydrogen density factor = " << denfactor << endl;

//Normalisation constant
double norm = (1.0 - alpha) / (pow(emax, 1.0 - alpha) - pow(emin, 1.0 - alpha));
cout << "Normalisation constant = " << norm << endl;

//Calculate Angle Binning levels
for (int i = 0; i < abins; i++)
{
azibinminlimits[i] = (azimin + (azimax-azimin) * i ) / abins;
azibinmaxlimits[i] = (azimin + (azimax-azimin) * (i+1) ) / abins;
polarbinminlimits[i] = ( (polarmin + (polarmax-polarmin)*i/abins) - 1);
polarbinmaxlimits[i] = ( (polarmin + (polarmax-polarmin)*(i+1)/abins) - 1);
}

//Populate max arrays for rejection sampling
for(int i = 0; i < ebins; ++i)
{
    double raymaxvalue = (DCS_Rayl(1, binenergy(0), 0.00000001))*twoPI*sin(0.00000001);
    double compmaxvalue = (DCS_Compt(1, binenergy(0), 0.00000001))*twoPI*sin(0.00000001);
    for(int j = 1; j < 181.0; ++j)
    {
        double rayvalue = (DCS_Rayl(1, binenergy(i), j*(myPI/180.0)))*twoPI*sin(j*(myPI
/180.0));
        if(rayvalue > raymaxvalue)
        {
            raymaxvalue = rayvalue;
        }
        double compvalue = (DCS_Compt(1, binenergy(i), j*(myPI/180.0)))*twoPI*sin(j*(myPI
/180.0));
        if(compvalue > compmaxvalue)
        {
            compmaxvalue = compvalue;
        }
    }
    raymaxes[i][0] = raymaxvalue*1.05;
    compmaxes[i][0] = compmaxvalue*1.05;
}

// Intialize all values in spectral array to 0
for(int i = 0; i < ebins; i++)
{
    for(int j = 0; j < abins; j++)
    {
        for(int k = 0; k < abins; k++)
        {
            spectra[i][j][k] = 0.0;
        }
    }
}

// Find cloud centre coords
double cloudcentrex = clouddistance * sin(cloudpolar) * cos(cloudazi);
double cloudcentrey = clouddistance * sin(cloudpolar) * sin(cloudazi);
double cloudcentrez = clouddistance * cos(cloudpolar);

for (double counter = 0; counter < iterations; ++counter)
{
    //Generate the photon
```

```
    double pe = photongen(alpha); //Photon energy
    double scatcount = 0.0; //Scatter counter
    bool inittest = false; //Bool for testing if photon is initial

    //Set photon position to origin
    double photonx = 0.0;
    double photony = 0.0;
    double photonz = 0.0;

    //Isotropically release photon in solid angle of cloud (angles are in photons frame)
    double dpolar = cloudadovertwo*randomnum();
    double dazi = twoPI*randomnum();

    //Convert dpsi and dtheta into direction vectors in absolute frame
    double youx = (sin(cloudpolar)*cos(cloudazi)*cos(dpolar)) + (sin(dpolar)*(cos(
cloudpolar)*cos(dazi)*cos(cloudazi) - sin(dazi)*sin(cloudazi)));
    double youy = (sin(cloudpolar)*sin(cloudazi)*cos(dpolar)) + (sin(dpolar)*(cos(
cloudpolar)*cos(dazi)*sin(cloudazi) + sin(dazi)*cos(cloudazi)));
    double youz = (cos(cloudpolar)*cos(dpolar)) - (sin(dpolar)*sin(cloudpolar)*cos(dazi))
;

    //Calculate photon direction angles in absolute frame
    double azi = atan2(youy, youx);
    if(azi < 0.0) //Account for negative output from atan2
    {
        azi = twoPI + azi;
    }
    double polar = acos(youz);

    //Find distance to edge of cloud
    double distancetocloudfromsource = spheredistanceoutside(photonx, photony, photonz,
cloudcentrex, cloudcentrey, cloudcentrez, azi, polar, clouddiameter);

    //If photon missed cloud, release new photon
    if (distancetocloudfromsource == 0.0)
    {
        continue;
    }

    //Set photon position to edge of cloud, adding a small amount to account for floating
 point errors
    photonx = photonx + (100000000000. + distancetocloudfromsource) * sin(polar) * cos(
azi);
    photony = photony + (100000000000. + distancetocloudfromsource) * sin(polar) * sin(
azi);
    photonz = photonz + (100000000000. + distancetocloudfromsource) * cos(polar);

    for(int counter2 = 0; counter2 < scatlimit; counter2++)
    {
        // Calculate distance to far side of cloud along photon direction vector
        double distancetofarsidecloud = spheredistanceinside(photonx, photony, photonz,
cloudcentrex, cloudcentrey, cloudcentrez, azi, polar, clouddiameter);

        //Find photon energy in joules
        double pej = kev2joules*pe;

        //Convert energy to wavelength
        double wavelength = (cbyh)/(pej);

        //Determine absorption column density encountered before absorption
        double nabs = -log(randomnum())/(totalcs(pe));;

        //Find Rayleigh and Compton cross sections
        double raycs = rayleighcs(pe);
        double comcs = comptoncs(pe);

        //Determine scattering column density encountered before scattering
        double nscat = -log(randomnum())/(raycs + comcs);

        // Determine the column density of the initial photon lines path through the
cloud
```

```
        double cdtodarsideofcloud = cloudhdensity*denfactor*distancetofarsidecloud;

        //Test for photons fate
        if ((nabs >= cdtodarsideofcloud) && (nscat >= cdtodarsideofcloud))
        {
            //Photon has escaped, test if its an initial photon
            if (inittest)
            {
      //Bin photon
      spectra[findenergybin(pe)][findazibin(azi, azibinminlimits, azibinmaxlimits)][
findpolarbin(polar, polarbinminlimits, polarbinmaxlimits)] += 1.0;
            }
            break;
        }
        else if (nabs <= nscat)
        {
            //Photon has been absorbed
            //Set new photon coords to point of absorption on initial photon line
            photonx = photonx + (nabs/(cloudhdensity*denfactor)) * sin(polar) * cos(azi);
            photony = photony + (nabs/(cloudhdensity*denfactor)) * sin(polar) * sin(azi);
            photonz = photonz + (nabs/(cloudhdensity*denfactor)) * cos(polar);

            if(pe >= ironkbinding)
            {
                //Introduce the fyield
                double ironalbedobyfyield = ((indycs(pe, 15) * (ironpercentagefactor))/(
totalcs(pe)))*fyield;

                if (randomnum() <= ironalbedobyfyield)
                {
                    //Re-emit photon isotropically
                    polar = acos((2.0*randomnum()) - 1.0);
                    azi = twoPI*randomnum();

                    //Test if re-emitted photon is k-alpha or k-beta
                    if(randomnum() <= kabratio)
                    {
                        pe = kbe; //Set photon energy to k-beta
                    }
                    else
                    {
                        pe = kae; //Set photon energy to k-alpha
                    }

                    inittest = true;
                    continue;
                }
            }
            break;
        }
        else
        {
            //Photon has been scattered
            //Stop multiple scattering if desired
            scatcount += 1;
            if((scatcount > 1) && (!multiplescattering))
            {
                break;
            }

            //Set new photon coords to point of scatter
            photonx = photonx + (nscat/(cloudhdensity*denfactor)) * sin(polar) * cos(azi)
;
            photony = photony + (nscat/(cloudhdensity*denfactor)) * sin(polar) * sin(azi)
;
            photonz = photonz + (nscat/(cloudhdensity*denfactor)) * cos(polar);

            double polarscat = 0.0;
            if (randomnum() <= (raycs/(raycs + comcs)))
            {
                //Photon Rayleigh scattered
```

```
                    polarscat = rayleighscatangle(pe);
                }
                else
                {
                    //Photon Compton scattered
                    polarscat = comptonscatangle(pe);
                    //Determine new energy of the photon
                    pe = (((cbyh)/( (homebyc) - ((homebyc)*cos(polarscat))  + (wavelength) ))
    /kev2joules);
                }
                double aziscat = twoPI*randomnum();

                //Convert psiscat and thetascat to direction vectors in absolute frame
                youx = (sin(polar)*cos(azi)*cos(polarscat)) + (sin(polarscat)*(cos(polar)*cos
    (aziscat)*cos(azi) - sin(aziscat)*sin(azi)));
                youy = (sin(polar)*sin(azi)*cos(polarscat)) + (sin(polarscat)*(cos(polar)*cos
    (aziscat)*sin(azi) + sin(aziscat)*cos(azi)));
                youz = (cos(polar)*cos(polarscat)) - (sin(polarscat)*sin(polar)*cos(aziscat))
    ;

        //Account for negative result of atan2
                azi = atan2(youy, youx);
                if(azi < 0.0)
                {
                    azi = twoPI + azi;
                }
                polar = acos(youz);

                inittest = true;
                continue;
            }
        }
    }

  //Output code run time
    time(&mid);
    cout << "Timemid = " << difftime(mid,start) << " s\n";

    // Print results to file
    string outfilestart = argv[6];
    string outname = outfilestart + ".txt";

  //Write to output file
    fstream outfile;
    outfile.open(outname.c_str(), ios::out);
    for (int i = 0; i < ebins; i++)
    {
        outfile << binenergy(i) << "\t";
        for (int j = 0; j < abins; j++)
        {
            for (int k = 0; k < abins; k++)
            {
                outfile << (norm*(spectra[i][j][k])) << "\t";
            }
        }
        outfile << "\n";
    }
    outfile.close();

    //Calculate & output miss %
    double per = ((double)misstest/(double)iterations) * 100;
    cout << "Miss test = " << misstest << " as % of total = " << per << endl;

    //Stop clock & output total elapsed time
    time(&finish);
    cout << "Time = " << difftime(finish,start) << " s" << endl;

  //End execution
    return 0;
}
```

```
double randomnum()
{
    return (double)rand()/(double)RAND_MAX;
}

int findazibin(double angle, double azimins[], double azimaxs[])
{
  for (int i = 0; i < abins; i++)
  {
        if ((angle >= azimins[i]) && (angle < azimaxs[i]))
        {
      return i;
        }
    }
    return 0;
}

int findpolarbin(double angle, double polarmins[], double polarmaxs[])
{
    angle = -1.0*cos(angle);
  for (int i = 0; i < abins; i++)
  {
        if ((angle >= polarmins[i]) && (angle <= polarmaxs[i]))
        {;
      return i;
        }
    }
    return 0;
}

double photongen(double alpha)
{
    return ( (emin) * pow( ( 1.0 + ( (randomnum()) *  ( pow( (emax/emin), (1.0-alpha) ) - 1.0
     ) ) ), (1.0 / (1.0-alpha)) ) );
}

int findenergybin(double photonenergy)
{
    if(photonenergy < 1.0)
        return 0;
    else
        return int(round((ebins*(log10(photonenergy/binmin)/log10(binmax/binmin)))));
}

double binenergy(int bin)
{
    return binmin*pow( (binmax/binmin), (double(bin)/ebinsf) );
}

double indycs(double pe, int ele)
{
    double delta;
    double csvalue = 0.0;

    for(int i = 0; i < 75; ++i)
    {
        if(pe > crossenergy[i])
        {
            continue;
        }
        else
        {
            delta = crossenergy[i] - crossenergy[i-1];
            csvalue = (crossvalues[ele][i] * barn * ((pe - crossenergy[i-1])/delta)) + (
    crossvalues[ele][i-1] * barn * ((crossenergy[i] - pe)/delta));
            break;
        }
    }
    return csvalue;
}
```

```
double totalcs(double pe)
{
    double cs = 0.0;
    for(int i = 0; i < 17; ++i)
    {
       cs +=  ( indycs(pe, i) * ( 1 / pow( (10), (met[0] - met[i]) ) ) );
    }
    return cs;
}

double rayleighcs(double pe)
{
    static double heliumfactor = (1/pow( (10), (met[0] - met[1])));
    return ((4.0*CSb_Rayl(1, pe)) + (CSb_Rayl(2, pe)*heliumfactor))*barn; //H + He
}

double comptoncs(double pe)
{
    static double heliumfactor = (1/pow( (10), (met[0] - met[1])));
    return ((2.0*CSb_Compt(1, pe)) + (CSb_Compt(2, pe)*heliumfactor))*barn; //H + He
}

double rayleighscatangle(double photonenergy)
{
    bool ok = false;
    double testangle;
    while (!ok)
    {
        testangle = myPI*randomnum();
        double rayds = ( (DCS_Rayl(1, photonenergy, testangle)*sin(testangle)) / (raymaxes[
    findenergybin(photonenergy)][0]) );
        if(randomnum() <= rayds)
            ok = true;
    }
    return testangle;
}

double comptonscatangle(double photonenergy)
{
    bool ok = false;
    double testangle;
    while (!ok)
    {
        testangle = myPI*randomnum();
        double comds = ( (DCS_Compt(1, photonenergy, testangle)*twoPI*sin(testangle)) / (
    compmaxes[findenergybin(photonenergy)][0]) );
        if(randomnum() <= comds)
            ok = true;
    }
    return testangle;
}

double distancebetweentwopoints(double x1, double y1, double z1, double x2, double y2, double
     z2)
{
    return sqrt(((x2-x1)*(x2-x1))+((y2-y1)*(y2-y1))+((z2-z1)*(z2-z1)));
}

double spheredistanceoutside(double px, double py, double pz, double cx, double cy, double cz
    , double azi, double polar, double clouddiameter)
{
    double t = 0.0, tp = 0.0, tn = 0.0;
    double x2, y2, z2;
    static double radius = clouddiameter/2.0;

    double px2 = px + (((clouddiameter)/4.0)*sin(polar)*cos(azi));
    double py2 = py + (((clouddiameter)/4.0)*sin(polar)*sin(azi));
    double pz2 = pz + (((clouddiameter)/4.0)*cos(polar));

    double dx1 = (px2-px);
    double dy1 = (py2-py);
```

```
    double dz1 = (pz2-pz);

    double dx2 = (px-cx);
    double dy2 = (py-cy);
    double dz2 = (pz-cz);

    double a = (dx1*dx1 + dy1*dy1 + dz1*dz1);
    double b = 2*( dx1*dx2 + dy1*dy2 + dz1*dz2 );
    double c = ((cx*cx) + (cy*cy) + (cz*cz) + (px*px) + (py*py) + (pz*pz) - (2*((cx*px)+(cy*
    py)+(cz*pz)))) - (radius*radius);

    double disc = ((b*b) - (4*a*c));
    if (disc < 0.0)
    {
        //No Intersection
        misstest += 1;
        cout << "Missed" << endl;
        return 0;
    }
    else if (disc == 0.0)
    {
        //Tangent
        t = -b/(2*a);
        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);
        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else if (disc > 0.0)
    {
        //Two Points of intersection
        tp = (-b + sqrt(disc) )/(2*a);
        tn = (-b - sqrt(disc) )/(2*a);

        // Make sure tp is smaller than tn
        if (tp > tn)
        {
            // If tp is bigger than tn swap them around
            double temp = tp;
            tp = tn;
            tn = temp;
        }
        // If tn is less than zero, the point is in the ray's negative direction
        if (tn < 0)
        {
            t = tp;
        }
        // If tp is less than zero, the intersection point is at tn
        if (tp < 0)
        {
            t = tn;
        }
        else // Else the intersection point is at t0
        {
            t = tp;
        }

        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);
        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else
    {
        cout << "Unknown Error" << endl;
        return 0;
    }

}
```

```
double spheredistanceinside(double px, double py, double pz, double cx, double cy, double cz,
      double azi, double polar, double clouddiameter)
{
    double t = 0.0, tp = 0.0, tn = 0.0;
    double x2, y2, z2;
    static double radius = clouddiameter/2.0;

    double px2 = px + ((clouddiameter/2.0)*sin(polar)*cos(azi));
    double py2 = py + ((clouddiameter/2.0)*sin(polar)*sin(azi));
    double pz2 = pz + ((clouddiameter/2.0)*cos(polar));

    double dx1 = (px2-px);
    double dy1 = (py2-py);
    double dz1 = (pz2-pz);

    double dx2 = (px-cx);
    double dy2 = (py-cy);
    double dz2 = (pz-cz);

    double a = (dx1*dx1 + dy1*dy1 + dz1*dz1);
    double b = 2*( dx1*dx2 + dy1*dy2 + dz1*dz2 );
    double c = (cx*cx) + (cy*cy) + (cz*cz) + (px*px) + (py*py) + (pz*pz) - (2*(cx*px+cy*py+cz
    *pz)) - (radius*radius);

    double disc = ((b*b) - (4*a*c));
    if (disc < 0.0)
    {
        //No Intersection
        return 0;
    }
    else if (disc == 0.0)
    {
        //Tangent
        t = -b/(2*a);
        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);

        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else if (disc > 0.0)
    {
        //Two points of intersection
        tp = (-b + sqrt(disc) )/(2*a);
        tn = (-b - sqrt(disc) )/(2*a);

        // If tn is less than zero, the point is in the ray's negative direction
        if (tp < 0.0)
        {
            t = tn;
        }
        else if (tn < 0.0)
        {
            t = tp;
        }

        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);

        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else
    {
        cout << "Unknown Error" << endl;
        return 0;
    }

}

double densityfactor(double density)
```

```
{
    double ne = 0.0;
    for(int i = 0; i < 17; i++)
    {
        ne += pow((10), (met[i]-12)) * density;
    }
    return ne;
}
```

Listing B.1: Full Monte Carlo Code - Uniform density

B.1.2 Non-Uniform Density Code

Snippet B.1 shows the Monte Carlo code in its entirety, the variable density version. See Chapter 2 for summary of operation.

```
#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <string>
#include <fstream>
#include <ctime>
#include <iostream>
#include "xraylib.h"
using namespace std;

//Constants
//Physical Constants
const double myPI = atan(1) * 4.0;
const double twoPI = atan(1) * 8.0;
const double Plankconst = 6.62606957E-34;
const double Massofelectron = 9.10938291E-31;//kg
const double speedoflight = 299792458; //m/s
const double speedoflightkn = 2.99792458e10; //cm/s
const double parsec = 3.08567758e18; //cm
const double ironkbinding = 7.114; //kev
const double fyield = 0.34; // Iron kalpha fluoresecense yield
const double electronradius = 2.8179403267e-13; // cm
const double electronradiuskn = 2.8179403267e-15; // m
const double kae = 6.399; // K alpha emission energy kev
const double kbe = 7.05; //K beta emission energy kev
const double kabratio = 0.13; // Ratio of k-alpha to k-beta emission
const double barn = 1E-24;   // cm^2
const double kev2joules = 1.602176E-16;

//Code Specific Constants
const double scatlimit = 35;    //Hard limit to number of successive scatters
const bool multiplescattering = true; //Are photons aloud to multiply scatter up to scatter
    limit
const int ebins = 400; //Number of energy bins
const float ebinsf = 400.0;
const double emin = 1.0; //keV
const double emax = 400.0; //keV
const double binmin = 1.0; //keV
const double binmax = 400.0; //keV
const int abins = 360; //Number of angular bins
const double azimin = 0.0; //Theta bin min
const double azimax = twoPI; //Theta bin max
const double polarmin = 0.0; //Polar bin min in cosine
const double polarmax = 2.0; //Polar bin max in cosine
const double cloudazi = 0.0; //0.610865238 //1.57079633 //2.35619449
const double cloudpolar = (myPI/2.0);
const double clouddistance = (25.0*parsec);//To centre of cloud //cm
const double observerdistance = (840.0*parsec);//cm

//Gaussian Variables
```

```
const double expectedvalue = 0.0; //Expected value, position of peak centre
const double stdev = 2.75; //standard deviation
const double d = 0; //Asymptotical value

//Functions
double randomnum();//Return a random number 0<x<1
double photongen(double alpha);//Generates a photon from power law
int findenergybin(double photonenergy);//Returns appropriate bin of input energy
double binenergy(int bin);//Determine energy of input bin
double rayleighscatangle(double photonenergy); //Return a Rayleigh scattering angle
double comptonscatangle(double photonenergy); //Return a Compton scattering angle
double rayleighcs(double pe); //Rayleigh scattering cross section
double comptoncs(double pe); //Compton scattering cross section
double indycs(double pe, int pos);
double totalcs(double pe);
double spheredistanceoutside(double px, double py, double pz, double cx, double cy, double cz
    , double azi, double polar, double cloudsize); //Ray trace sphere
double spheredistanceinside(double px, double py, double pz, double cx, double cy, double cz,
     double azi, double polar, double cloudsize); //Ray trace sphere
double distancebetweentwopoints(double x1, double y1, double z1, double x2, double y2, double
     z2);
double dotproduct(double x1, double y1, double z1, double x2, double y2, double z2);
double frsq(double radius);
double densityfactor(double density);
int findazibin(double angle, double azimins[], double azimaxs[]);//Returns the angle bin
int findpolarbin(double angle, double polarmins[], double polarmaxs[]); //Returns angle bin

//Cross Section value holding arrays
double crossenergy[75];//Energies for different cross sections
double crossvalues[17][75];//Array holding various element cross sections;
double met[17];//Array holding elemental abundances
string crossfiles[19] = {"cs3/H.txt","cs3/He.txt","cs3/C.txt","cs3/N.txt","cs3/O.txt","cs3/Ne
    .txt","cs3/Na.txt","cs3/Mg.txt","cs3/Al.txt","cs3/Si.txt","cs3/S.txt","cs3/Cl.txt","cs3/
    Ar.txt","cs3/Ca.txt","cs3/Cr.txt","cs3/Fe.txt","cs3/Ni.txt","cs3/abundances.txt","cs3/
    energies_1-600_kev.txt"};

//Calculate Angle Binning levels
double azibinminlimits[abins];
double azibinmaxlimits[abins];
double polarbinminlimits[abins];
double polarbinmaxlimits[abins];
//Max arrays
double raymaxes[ebins][1];
double compmaxes[ebins][1];
//Spectra Array
float spectra[ebins][abins][abins];//Output spectrum

//Number of misses
int misstest = 0;

int main(int argc, char *argv[])
{
    //Intialise RNG
    srand(2658479);

    //Start clock
    time_t start, mid, finish;
    time(&start);

    if(argc != 9)
    {
        cout << "Wrong number of parameters, Exiting program.\nCorrect Entry is:\n1)
    Iterations, 2)Alpha, 3)Cloud central Density, 4)Density profile (1 = 1/r^2, 2 = e^-r, 3 =
     gaussian), 5)Stepsize (parsecs), 6)Cloud Diamater, 7)Iron abundance as factor of Solar,
    8)Output file name." << endl;
        return 0;
    }

    //Command line arguemnts
    double iterations = atof(argv[1]);
    double alpha = atof(argv[2]);
```

```
  double cloudhdensity = atof(argv[3]);
  int densitychoice = atoi(argv[4]);
  double stepsize = atof(argv[5]);
  double stepsizecm = stepsize*parsec;
  double clouddiameter = atof(argv[6])*parsec;
  double ironabundancefactor = atof(argv[7]);

  cout << "Alpha = " << alpha << endl;
  cout << "Iterations = " << iterations << endl;

  // Calculate needed values
  // Cloud radius
  double cloudradius = (clouddiameter)/2.0;
  cout << "Cloud radius = " << (cloudradius/parsec) << endl;
  // Cloud angular diameter from source
  double cloudad = 2 * asin( (clouddiameter)/(2*(clouddistance)) );
  double cloudadovertwo = cloudad/2.0;
  cout << "Cloud angular diamter = " << cloudad << " Radians & " << (cloudad*(180/myPI)) <<
    " Degrees" << endl;

  //Access and bin abundances & cross sections;
//Metallicities
  fstream infile;
  infile.open (crossfiles[17].c_str(), ios::in);
  int metplace = 0;
  for(int i = 0; i < 34; ++i)
  {
      string metinput;
      getline(infile,metinput);
      if(metinput[0] == '#')
      {
          continue;
      }
      else
      {
          met[metplace] = atof(metinput.c_str());
          ++metplace;
      }
  }
  infile.close();
//Energy values
  infile.open (crossfiles[18].c_str(), ios::in);
  for(int i = 0; i < 75; ++i)
  {
      infile >> crossenergy[i];
  }
  infile.close();

//Raise Iron abundance by user set amount
  cout << "Solar Fe abundance " << met[15]<< endl;
  met[15] = log10( ironabundancefactor * pow( 10.0,(met[15] - 12.0) ) ) + 12.0;
  cout << "Cloud Fe Abundance = " << met[15] << endl;

//Elemental cross sections
  for(int i = 0; i < 17; ++i)
  {
      infile.open (crossfiles[i].c_str(), ios::in);
      for(int j = 0; j < 75; ++j)
      {
          infile >> crossvalues[i][j];
      }
      infile.close();
  }

  // Working constants
  // Iron abundance %
  double ironpercentagefactor =  1.0/pow( (10), (met[0] - met[15]));
  // Speed of light by planck constant
  double cbyh = speedoflight*Plankconst;
  // Planck constant over mass of electron times speed of light
  double homebyc = Plankconst/(Massofelectron*speedoflight);
```

```
//Factor to add contribution from non-hydrogen atoms
double denfactor = densityfactor(cloudhdensity)/cloudhdensity;
cout << "Non-Hydrogen density factor = " << denfactor << endl;

//Normalisation constant
double norm = (1.0 - alpha) / (pow(emax, 1.0 - alpha) - pow(emin, 1.0 - alpha));
cout << "Normalisation constant = " << norm << endl;

//Calculate Angle Binning levels
for (int i = 0; i < abins; i++)
{
azibinminlimits[i] = (azimin + (azimax-azimin) * i ) / abins;
azibinmaxlimits[i] = (azimin + (azimax-azimin) * (i+1) ) / abins;
polarbinminlimits[i] = ( (polarmin + (polarmax-polarmin)*i/abins) - 1);
polarbinmaxlimits[i] = ( (polarmin + (polarmax-polarmin)*(i+1)/abins) - 1);
}

  //Populate max arrays for rejection sampling
for(int i = 0; i < ebins; ++i)
{
    double raymaxvalue = (DCS_Rayl(1, binenergy(0), 0.00000001))*twoPI*sin(0.00000001);
    double compmaxvalue = (DCS_Compt(1, binenergy(0), 0.00000001))*twoPI*sin(0.00000001);
    for(int j = 1; j < 181.0; ++j)
    {
        double rayvalue = (DCS_Rayl(1, binenergy(i), j*(myPI/180.0)))*twoPI*sin(j*(myPI
/180.0));
        if(rayvalue > raymaxvalue)
        {
            raymaxvalue = rayvalue;
        }
        double compvalue = (DCS_Compt(1, binenergy(i), j*(myPI/180.0)))*twoPI*sin(j*(myPI
/180.0));
        if(compvalue > compmaxvalue)
        {
            compmaxvalue = compvalue;
        }
    }
    raymaxes[i][0] = raymaxvalue*1.05;
    compmaxes[i][0] = compmaxvalue*1.05;
}
// Intialize all values in spectral array to 0
for(int i = 0; i < ebins; i++)
{
    for(int j = 0; j < abins; j++)
    {
        for(int k = 0; k < abins; k++)
        {
            spectra[i][j][k] = 0.0;
        }
    }
}

// Find cloud centre coords
double cloudcentrex = clouddistance * sin(cloudpolar) * cos(cloudazi);
double cloudcentrey = clouddistance * sin(cloudpolar) * sin(cloudazi);
double cloudcentrez = clouddistance * cos(cloudpolar);

for (double counter = 0; counter < iterations; ++counter)
{
    //Generate the photon
    double pe = photongen(alpha); //Photon energy
    double scatcount = 0.0; //Scatter counter
    bool inittest = false; //Bool for testing if photon is initial

    //Set photon position to origin
    double photonx = 0.0;
    double photony = 0.0;
    double photonz = 0.0;

    //Isotropically release photon in solid angle of cloud (angles are in photons frame)
    double dpolar = cloudadovertwo*randomnum();
```

```
    double dazi = twoPI*randomnum();

    //Convert dpsi and dtheta into direction vectors in absolute frame
    double youx = (sin(cloudpolar)*cos(cloudazi)*cos(dpolar)) + (sin(dpolar)*(cos(
cloudpolar)*cos(dazi)*cos(cloudazi) - sin(dazi)*sin(cloudazi)));
    double youy = (sin(cloudpolar)*sin(cloudazi)*cos(dpolar)) + (sin(dpolar)*(cos(
cloudpolar)*cos(dazi)*sin(cloudazi) + sin(dazi)*cos(cloudazi)));
    double youz = (cos(cloudpolar)*cos(dpolar)) - (sin(dpolar)*sin(cloudpolar)*cos(dazi))
;

    //Calculate photon direction angles in absolute frame
    double azi = atan2(youy, youx);
    if(azi < 0.0) //Account for negative output from atan2
    {
        azi = twoPI + azi;
    }
    double polar = acos(youz);

    //Find distance to edge of cloud
    double distancetocloudfromsource = spheredistanceoutside(photonx, photony, photonz,
cloudcentrex, cloudcentrey, cloudcentrez, azi, polar, clouddiameter);

    //If photon missed cloud, release new photon
    if (distancetocloudfromsource == 0.0)
    {
        continue;
    }

    //Set photon position to edge of cloud, adding a small amount to account for floating
 point errors
    photonx = photonx + (100000000000. + distancetocloudfromsource) * sin(polar) * cos(
azi);
    photony = photony + (100000000000. + distancetocloudfromsource) * sin(polar) * sin(
azi);
    photonz = photonz + (100000000000. + distancetocloudfromsource) * cos(polar);

    for(int counter2 = 0; counter2 < scatlimit; counter2++)
    {
        //Find photon energy in joules
        double pej = kev2joules*pe;

        //Convert energy to wavelength
        double wavelength = (cbyh)/(pej);

        //Determine absorption column density encountered before absorption
        double nabs = -log(randomnum())/(totalcs(pe));
 //Find Rayleigh and Compton cross sections
        double raycs = rayleighcs(pe);
        double comcs = comptoncs(pe);
        //Determine scattering column density encountered before scattering
        double nscat = -log(randomnum())/(raycs + comcs);

        int flag;
        double cd = 0.0;
        double distancetrav = 0.0;
        if (nscat < nabs)
        {
            //It will either scatter or escape
            flag = 1; //Scatter

            while (cd < nscat)
            {
                double radiusparsecs = distancebetweentwopoints(photonx, photony, photonz
, cloudcentrex, cloudcentrey, cloudcentrez)/parsec;
                switch(densitychoice)
                {
                    case 1:
                        cd += stepsizecm*cloudhdensity*frsq(radiusparsecs);
                        break;

                    case 2:
```

```
                        cd += stepsizecm*exp(-radiusparsecs)*cloudhdensity;
                        break;

                    case 3:
                        cd += stepsizecm*cloudhdensity*exp(-(((radiusparsecs-
expectedvalue)*(radiusparsecs-expectedvalue))/(2*stdev*stdev))) + d;
                        break;
                }

                distancetrav += stepsize;

                //Propagate photon by the stepsize
                photonx = photonx + stepsizecm*sin(polar)*cos(azi);
                photony = photony + stepsizecm*sin(polar)*sin(azi);
                photonz = photonz + stepsizecm*cos(polar);

                if ((distancebetweentwopoints(photonx, photony, photonz, cloudcentrex,
cloudcentrey, cloudcentrez)) > cloudradius)
                {
                    flag = 3; // Escaped
                    break;
                }

            }

        }
        else //Absorb or escape
        {
            flag = 2; // It will either absorb or escape

            while (cd < nabs)
            {
                double radiusparsecs = distancebetweentwopoints(photonx, photony, photonz
, cloudcentrex, cloudcentrey, cloudcentrez)/parsec;
                switch(densitychoice)
                {
                    case 1:
                        cd += stepsizecm*cloudhdensity*frsq(radiusparsecs);
                        break;

                    case 2:
                        cd += stepsizecm*exp(-radiusparsecs)*cloudhdensity;
                        break;

                    case 3:
                        cd += stepsizecm*cloudhdensity*exp(-(((radiusparsecs-
expectedvalue)*(radiusparsecs-expectedvalue))/(2*stdev*stdev))) + d;
                        break;
                }

                distancetrav += stepsize;

                //Propagate photon by the stepsize
                photonx = photonx + stepsizecm*sin(polar)*cos(azi);
                photony = photony + stepsizecm*sin(polar)*sin(azi);
                photonz = photonz + stepsizecm*cos(polar);

                if ((distancebetweentwopoints(photonx, photony, photonz, cloudcentrex,
cloudcentrey, cloudcentrez)) > cloudradius)
                {
                    flag = 3; // Escaped
                    break;
                }

            }
        }

        //Test for photons fate
        if (flag == 3)
        {
            //Photon has escaped
```

```
            if (inittest)
            {
                //Convert angles greater than pi, to the equivalent in 0<x<pi
                if (azi > myPI)
                {
                    azi = twoPI - azi;
                    spectra[findenergybin(pe)][findazibin(azi, azibinminlimits,
azibinmaxlimits)][findpolarbin(polar, polarbinminlimits, polarbinmaxlimits)] += 1.0;
                }
                else
                {
                    spectra[findenergybin(pe)][findazibin(azi, azibinminlimits,
azibinmaxlimits)][findpolarbin(polar, polarbinminlimits, polarbinmaxlimits)] += 1.0;
                }
            }
            break;
        }
        else if (flag == 2)
        {
            //Photon has been absorbed
            if(pe >= ironkbinding)
            {
                //Introduce the fyield
                double ironalbedobyfyield = ((indycs(pe, 15) * (ironpercentagefactor))/(
totalcs(pe)))*fyield;

                if (randomnum() <= ironalbedobyfyield)
                {
                    //Re-emit photon isotropically
                    polar = acos((2.0*randomnum()) - 1.0);
                    azi = twoPI*randomnum();

                    //Test if re-emitted photon is k-alpha or k-beta
                    if(randomnum() <= kabratio)
                    {
                        pe = kbe; //Set photon energy to k-beta
                    }
                    else
                    {
                        pe = kae; //Set photon energy to k-alpha
                    }

                    inittest = true;
                    continue;
                }
            }
            break;
        }
        else
        {
            //Photon has been scattered
            //Stop multiple scattering if desired
            scatcount += 1;
            if((scatcount > 1) && (!multiplescattering))
            {
                break;
            }

            double polarscat = 0.0;
            if (randomnum() <= (rayleighcs(pe)/(rayleighcs(pe) + comptoncs(pe))))
            {
                //Photon Rayleigh scattered
                polarscat = rayleighscatangle(pe);
            }
            else
            {
                //Photon Compton scattered
                polarscat = comptonscatangle(pe);
                //Determine new energy of the photon
                pe = (((cbyh)/( (homebyc) - ((homebyc)*cos(polarscat))  + (wavelength) ))
/kev2joules);
```

```
                }
                double aziscat = twoPI*randomnum();

                //Convert psiscat and thetascat to direction vectors in absolute frame
                youx = (sin(polar)*cos(azi)*cos(polarscat)) + (sin(polarscat)*(cos(polar)*cos
    (aziscat)*cos(azi) - sin(aziscat)*sin(azi)));
                youy = (sin(polar)*sin(azi)*cos(polarscat)) + (sin(polarscat)*(cos(polar)*cos
    (aziscat)*sin(azi) + sin(aziscat)*cos(azi)));
                youz = (cos(polar)*cos(polarscat)) - (sin(polarscat)*sin(polar)*cos(aziscat))
    ;

                azi = atan2(youy, youx);
                if(azi < 0.0)
                {
                    azi = twoPI + azi;
                }
                polar = acos(youz);

                //Determine new energy of photon
                pe = (((cbyh)/( (homebyc) - ((homebyc)*cos(polarscat))  + (wavelength) ))/
    kev2joules);

                inittest = true;
                continue;
            }
        }
    }

  //Output code run time
    time(&mid);
    cout << "Timemid = " << difftime(mid,start) << " s\n";

    // Print results to file
    string outfilestart = argv[8];
    string outname = outfilestart + ".txt";

  //Write to output file
    fstream outfile;

    outfile.open(outname.c_str(), ios::out);
    for (int i = 0; i < ebins; i++)
    {
        outfile << binenergy(i) << "\t";
        for (int j = 0; j < abins; j++)
        {
            for (int k = 0; k < abins; k++)
            {
                //outfile << ((norm*(spectra[i][j][k])/(binenergy(i+1) - binenergy(i)))) <<
    "\t";
                outfile << (norm*(spectra[i][j][k])) << "\t";
            }
        }
        outfile << "\n";
    }
    outfile.close();

    //Calculate & output miss %
    double per = ((double)misstest/(double)iterations) * 100;
    cout << "Miss test = " << misstest << " as % of total = " << per << endl;

    //Stop clock & output total elapsed time
    time(&finish);
    cout << "Time = " << difftime(finish,start) << " s" << endl;

    return 0;
}

double randomnum()
{
    return (double)rand()/(double)RAND_MAX;
}
```

```
int findazibin(double angle, double azimins[], double azimaxs[])
{
  for (int i = 0; i < abins; i++)
  {
        if ((angle >= azimins[i]) && (angle < azimaxs[i]))
        {
      return i;
        }
    }
    return 0;
}

int findpolarbin(double angle, double polarmins[], double polarmaxs[])
{
    angle = -1.0*cos(angle);
  for (int i = 0; i < abins; i++)
  {
        if ((angle >= polarmins[i]) && (angle <= polarmaxs[i]))
        {
      return i;
        }
    }
    return 0;
}

double photongen(double alpha)
{
    return ( (emin) * pow( ( 1.0 + ( (randomnum()) *  ( pow( (emax/emin), (1.0-alpha) ) - 1.0
      ) ) ), (1.0 / (1.0-alpha)) ) );
}

int findenergybin(double photonenergy)
{
    if(photonenergy < 1.0)
        return 0;
    else
        return int(round((ebins*(log10(photonenergy/binmin)/log10(binmax/binmin)))));
}

double binenergy(int bin)
{
    return binmin*pow( (binmax/binmin), (double(bin)/ebinsf) );
}

double rayleighcs(double pe)
{
    static double heliumfactor = (1/pow( (10), (met[0] - met[1])));
    return ((4.0*CSb_Rayl(1, pe)) + (CSb_Rayl(2, pe)*heliumfactor))*barn; //H + He
}

double comptoncs(double pe)
{
    static double heliumfactor = (1/pow( (10), (met[0] - met[1])));
    return ((2.0*CSb_Compt(1, pe)) + (CSb_Compt(2, pe)*heliumfactor))*barn; //H + He
}

double rayleighscatangle(double photonenergy)
{
    //Generate a rayleigh event angle
    bool ok = false;
    double testangle;
    while (!ok)
    {
        testangle = myPI*randomnum();
        double rayds = ( (DCS_Rayl(1, photonenergy, testangle)*sin(testangle)) / (raymaxes[
    findenergybin(photonenergy)][0]) );
        //cout << test << endl;
        if(randomnum() <= rayds)
            ok = true;
    }
```

```
    return testangle;
}

double comptonscatangle(double photonenergy)
{
    //Generate a compton event angle
    bool ok = false;
    double testangle;
    while (!ok)
    {
        testangle = myPI*randomnum();
        double comds = ( (DCS_Compt(1, photonenergy, testangle)*twoPI*sin(testangle)) / (
    compmaxes[findenergybin(photonenergy)][0]) );
        //cout << test << endl;
        if(randomnum() <= comds)
            ok = true;
    }
    return testangle;
}

double indycs(double pe, int ele)
{
    double delta;
    double csvalue = 0.0;

    for(int i = 0; i < 75; ++i)
    {
        if(pe > crossenergy[i])
        {
            continue;
        }
        else
        {
            delta = crossenergy[i] - crossenergy[i-1];
            csvalue = (crossvalues[ele][i] * barn * ((pe - crossenergy[i-1])/delta)) + (
    crossvalues[ele][i-1] * barn * ((crossenergy[i] - pe)/delta));
            break;
        }
    }
    return csvalue;
}

double totalcs(double pe)
{
    double cs = 0.0;
    for(int i = 0; i < 17; ++i)
    {
       cs +=  ( indycs(pe, i) * ( 1 / pow( (10), (met[0] - met[i]) ) ) );
    }
    return cs;
}

double distancebetweentwopoints(double x1, double y1, double z1, double x2, double y2, double
     z2)
{
    return sqrt(((x2-x1)*(x2-x1))+((y2-y1)*(y2-y1))+((z2-z1)*(z2-z1)));
}

double spheredistanceoutside(double px, double py, double pz, double cx, double cy, double cz
    , double azi, double polar, double clouddiameter)
{
    double t = 0.0, tp = 0.0, tn = 0.0;
    double x2, y2, z2;
    static double radius = clouddiameter/2.0;

    double px2 = px + (((clouddiameter)/4.0)*sin(polar)*cos(azi));
    double py2 = py + (((clouddiameter)/4.0)*sin(polar)*sin(azi));
    double pz2 = pz + (((clouddiameter)/4.0)*cos(polar));

    double dx1 = (px2-px);
    double dy1 = (py2-py);
```

```
    double dz1 = (pz2-pz);

    double dx2 = (px-cx);
    double dy2 = (py-cy);
    double dz2 = (pz-cz);

    double a = (dx1*dx1 + dy1*dy1 + dz1*dz1);
    double b = 2*( dx1*dx2 + dy1*dy2 + dz1*dz2 );
    double c = ((cx*cx) + (cy*cy) + (cz*cz) + (px*px) + (py*py) + (pz*pz) - (2*((cx*px)+(cy*
    py)+(cz*pz)))) - (radius*radius);

    double disc = ((b*b) - (4*a*c));
    if (disc < 0.0)
    {
        //No Intersection
        misstest += 1;
        cout << "Missed" << endl;
        return 0;
    }
    else if (disc == 0.0)
    {
        //Tangent
        t = -b/(2*a);
        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);
        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else if (disc > 0.0)
    {
        //Two Points of intersection
        tp = (-b + sqrt(disc) )/(2*a);
        tn = (-b - sqrt(disc) )/(2*a);

        // Make sure tp is smaller than tn
        if (tp > tn)
        {
            // If tp is bigger than tn swap them around
            double temp = tp;
            tp = tn;
            tn = temp;
        }
        // If tn is less than zero, the point is in the ray's negative direction
        if (tn < 0)
        {
            t = tp;
        }
        // If tp is less than zero, the intersection point is at tn
        if (tp < 0)
        {
            t = tn;
        }
        else // Else the intersection point is at t0
        {
            t = tp;
        }

        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);
        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else
    {
        cout << "Unknown Error" << endl;
        return 0;
    }

}
```

```
double spheredistanceinside(double px, double py, double pz, double cx, double cy, double cz,
      double azi, double polar, double clouddiameter)
{
    double t = 0.0, tp = 0.0, tn = 0.0;
    double x2, y2, z2;
    static double radius = clouddiameter/2.0;

    double px2 = px + ((clouddiameter/2.0)*sin(polar)*cos(azi));
    double py2 = py + ((clouddiameter/2.0)*sin(polar)*sin(azi));
    double pz2 = pz + ((clouddiameter/2.0)*cos(polar));

    double dx1 = (px2-px);
    double dy1 = (py2-py);
    double dz1 = (pz2-pz);

    double dx2 = (px-cx);
    double dy2 = (py-cy);
    double dz2 = (pz-cz);

    double a = (dx1*dx1 + dy1*dy1 + dz1*dz1);
    double b = 2*( dx1*dx2 + dy1*dy2 + dz1*dz2 );
    double c = (cx*cx) + (cy*cy) + (cz*cz) + (px*px) + (py*py) + (pz*pz) - (2*(cx*px+cy*py+cz
    *pz)) - (radius*radius);

    double disc = ((b*b) - (4*a*c));
    if (disc < 0.0)
    {
        //No Intersection
        return 0;
    }
    else if (disc == 0.0)
    {
        //Tangent
        t = -b/(2*a);
        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);

        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else if (disc > 0.0)
    {
        //Two points of intersection
        tp = (-b + sqrt(disc) )/(2*a);
        tn = (-b - sqrt(disc) )/(2*a);

        // If tn is less than zero, the point is in the ray's negative direction
        if (tp < 0.0)
        {
            t = tn;
        }
        else if (tn < 0.0)
        {
            t = tp;
        }

        x2 = px + (dx1*t);
        y2 = py + (dy1*t);
        z2 = pz + (dz1*t);

        return sqrt(  ((x2-px) * (x2-px)) + ((y2-py) * (y2-py)) + ((z2-pz) * (z2-pz))  );
    }
    else
    {
        cout << "Unknown Error" << endl;
        return 0;
    }

}

double densityfactor(double density)
```

```
{
    double ne = 0.0;
    for(int i = 0; i < 17; i++)
    {
        ne += pow((10), (met[i]-12)) * density;
    }
    return ne;
}

double dotproduct(double x1, double y1, double z1, double x2, double y2, double z2)
{
    return (x1*x2) + (y1*y2) + (z1*z2);
}

double frsq(double radius)
{
    if(radius < 1.0)
        return 1.0;
    else if (radius >= 1.0)
        return 1/(radius*radius);

    return 0;
}
```

Listing B.2: Full Monte Carlo Code - Variable density

B.1.3 Cylindrical Cloud Function

The function shown in snippet B.3 shows the function for calculating distances in the code when simulating a cylindrical cloud, designed to replace the function *spheredistance()* with this one in the code proper. See Section 2.2.6.

```
double cylinderdistance(double px, double py, double pz, double cenx, double ceny, double
    cenz, double angleazi, double anglepolar, double cloudradius, double cloudlength, double
    cylangleazi, double cylanglepolar)
{
    double candidates[4];
    for(int i = 0; i < 4; i++)
    {
        candidates[i] = 0.0;
    }

    double cx = cenx - ((cloudlength/2)*sin(cylanglepolar)*cos(cylangleazi));
    double cy = ceny - ((cloudlength/2)*sin(cylanglepolar)*sin(cylangleazi));
    double cz = cenz - ((cloudlength/2)*cos(cylanglepolar));

    double t = 0.0, tp = 0.0, tn = 0.0, t3 = 0.0, t4 = 0.0;

    double cx2 = cenx + ((cloudlength/2)*sin(cylanglepolar)*cos(cylangleazi));
    double cy2 = ceny + ((cloudlength/2)*sin(cylanglepolar)*sin(cylangleazi));
    double cz2 = cenz + ((cloudlength/2)*cos(cylanglepolar));

    double delpx = px - cx;
    double delpy = py - cy;
    double delpz = pz - cz;

    double raydirx = sin(anglepolar)*cos(angleazi);
    double raydiry = sin(anglepolar)*sin(angleazi);
    double raydirz = cos(anglepolar);

    double cyldirx = sin(cylanglepolar)*cos(cylangleazi);
    double cyldiry = sin(cylanglepolar)*sin(cylangleazi);
    double cyldirz = cos(cylanglepolar);

    //Calculating A
```

```
double smallax = cyldirx*dotproduct(raydirx, raydiry, raydirz, cyldirx, cyldiry, cyldirz)
;
double smallay = cyldiry*dotproduct(raydirx, raydiry, raydirz, cyldirx, cyldiry, cyldirz)
;
double smallaz = cyldirz*dotproduct(raydirx, raydiry, raydirz, cyldirx, cyldiry, cyldirz)
;

double medax = raydirx - smallax;
double meday = raydiry - smallay;
double medaz = raydirz - smallaz;

double a = dotproduct(medax, meday, medaz, medax, meday, medaz);

//Calculating B
double smallbx = cyldirx*dotproduct(delpx, delpy, delpz, cyldirx, cyldiry, cyldirz);
double smallby = cyldiry*dotproduct(delpx, delpy, delpz, cyldirx, cyldiry, cyldirz);
double smallbz = cyldirz*dotproduct(delpx, delpy, delpz, cyldirx, cyldiry, cyldirz);

double medbx = delpx - smallbx;
double medby = delpy - smallby;
double medbz = delpz - smallbz;

double b = 2*dotproduct(medax, meday, medaz, medbx, medby, medbz);

// Calculating C
double c = dotproduct(medbx, medby, medbz, medbx, medby, medbz);
c = c - (cloudradius*cloudradius);

double disc = ((b*b) - (4*a*c));
if (disc < 0.0)
{
    //No points of intersection
    misstest = misstest + 1;
    return 0.0;
}
else if (disc == 0.0)
{
    //Point of intersection is tangent
    t = -b/(2*a);
    double px2 = px + (raydirx*t);
    double py2 = py + (raydiry*t);
    double pz2 = pz + (raydirz*t);
    return sqrt(  ((px2-px) * (px2-px)) + ((py2-py) * (py2-py)) + ((pz2-pz) * (pz2-pz))
);
}
else if (disc > 0.0)
{
    //Two points of intersection
    tp = (-b + sqrt(disc) )/(2*a);
    tn = (-b - sqrt(disc) )/(2*a);

    //Testing for intersection with finite cylinder
    if (tp > 0.0)
    {
        double tptestonesmallx = px + (tp*sin(anglepolar)*cos(angleazi));
        double tptestonesmally = py + (tp*sin(anglepolar)*sin(angleazi));
        double tptestonesmallz = pz + (tp*cos(anglepolar));

        double tptestonemedx = tptestonesmallx - cx;
        double tptestonemedy = tptestonesmally - cy;
        double tptestonemedz = tptestonesmallz - cz;

        double tptestoneeq = dotproduct(cyldirx, cyldiry, cyldirz, tptestonemedx,
tptestonemedy, tptestonemedz);

        if (tptestoneeq > 0.0)
        {
            double tptesttwomedx = tptestonesmallx - cx2;
            double tptesttwomedy = tptestonesmally - cy2;
            double tptesttwomedz = tptestonesmallz - cz2;
```

```
            double tptesttwoeq = dotproduct(cyldirx, cyldiry, cyldirz, tptesttwomedx,
tptesttwomedy, tptesttwomedz);

            if (tptesttwoeq < 0.0)
            {
                if(tp > 0.0)
                {
                    candidates[0] = tp;
                }
            }
        }
    }

    if (tn > 0.0)
    {
        double tntestonesmallx = px + (tn*sin(anglepolar)*cos(angleazi));
        double tntestonesmally = py + (tn*sin(anglepolar)*sin(angleazi));
        double tntestonesmallz = pz + (tn*cos(anglepolar));

        double tntestonemedx = tntestonesmallx - cx;
        double tntestonemedy = tntestonesmally - cy;
        double tntestonemedz = tntestonesmallz - cz;

        double tntestoneeq = dotproduct(cyldirx, cyldiry, cyldirz, tntestonemedx,
tntestonemedy, tntestonemedz);

        if (tntestoneeq > 0.0)
        {
            double tntesttwomedx = tntestonesmallx - cx2;
            double tntesttwomedy = tntestonesmally - cy2;
            double tntesttwomedz = tntestonesmallz - cz2;

            double tntesttwoeq = dotproduct(cyldirx, cyldiry, cyldirz, tntesttwomedx,
tntesttwomedy, tntesttwomedz);

            if (tntesttwoeq < 0.0)
            {
                if(tn > 0.0)
                {
                    candidates[1] = tn;
                }
            }
        }
    }

    if ((tp < 0.0) && (tn < 0.0))
    {
        cout << "Whoops something went wrong\n";
    }

    //Testing for intersection with end caps of finite cylinder
    double t3testsmallx = cx - px;
    double t3testsmally = cy - py;
    double t3testsmallz = cz - pz;

    double t3testmed = dotproduct(cyldirx, cyldiry, cyldirz, t3testsmallx, t3testsmally,
t3testsmallz);

    t3 = t3testmed/(dotproduct(raydirx, raydiry, raydirz, cyldirx, cyldiry, cyldirz));

    double t4testsmallx = cx2 - px;
    double t4testsmally = cy2 - py;
    double t4testsmallz = cz2 - pz;

    double t4testmed = dotproduct(cyldirx, cyldiry, cyldirz, t4testsmallx, t4testsmally,
t4testsmallz);

    t4 = t4testmed/(dotproduct(raydirx, raydiry, raydirz, cyldirx, cyldiry, cyldirz));

    if(t3 > 0.0)
    {
```

```
        double intytestonex = px + (t3*sin(anglepolar)*cos(angleazi));
        double intytestoney = py + (t3*sin(anglepolar)*sin(angleazi));
        double intytestonez = pz + (t3*cos(anglepolar));

        double intytestonemedx = intytestonex - cx;
        double intytestonemedy = intytestoney - cy;
        double intytestonemedz = intytestonez - cz;

        double intytestone = dotproduct(intytestonemedx, intytestonemedy, intytestonemedz
, intytestonemedx, intytestonemedy, intytestonemedz);

        if(intytestone < (cloudradius*cloudradius))
        {
            if(t3 > 0.0)
            {
                candidates[2] = t3;
            }
        }
    }

    if(t4 > 0.0)
    {
        double intytestonex = px + (t4*sin(anglepolar)*cos(angleazi));
        double intytestoney = py + (t4*sin(anglepolar)*sin(angleazi));
        double intytestonez = pz + (t4*cos(anglepolar));

        double intytestonemedx = intytestonex - cx2;
        double intytestonemedy = intytestoney - cy2;
        double intytestonemedz = intytestonez - cz2;

        double intytestone = dotproduct(intytestonemedx, intytestonemedy, intytestonemedz
, intytestonemedx, intytestonemedy, intytestonemedz);

        if(intytestone < (cloudradius*cloudradius))
        {
            //t4 is candidate
            if(t4 > 0.0)
            {
                candidates[3] = t4;
            }
        }
    }

    double mint = 0.0;

    for(int f = 0; f < 4; f++)
    {
        if(candidates[f] > 0.0)
        {
            mint = candidates[f];
        }
    }

    for(int f = 1; f < 4; f++)
    {
        if(candidates[f] < mint )
        {
            if(candidates[f] > 0.0)
            {
                mint = candidates[f];
            }
        }
    }

    t = mint;

    double px2 = px + (t*sin(anglepolar)*cos(angleazi));
    double py2 = py + (t*sin(anglepolar)*sin(angleazi));
    double pz2 = pz + (t*cos(anglepolar));
```

```
        return sqrt(  ((px2-px) * (px2-px)) + ((py2-py) * (py2-py)) + ((pz2-pz) * (pz2-pz))
    );
    }
    else
    {
        cout << "Unknown Error" << endl;
        return 0;
    }
}
```

Listing B.3: Cylinder Function

B.2 Spectral Extraction Script

Listing B.4 shows the python script used to extract a single spectrum from the code output.

```
import sys
import math as m

if len(sys.argv) != 5:
  print "Wrong number of arguments, Input as follows: 1)Input File, 2)Cloud Position (degrees
      (0 - 180)), 3)Observer size (parsecs), 4)Output File."
  sys.exit()

inputfile = open(sys.argv[1], 'r')
outputfile = open(sys.argv[4], 'w')
angle = m.radians(float(sys.argv[2]))
observersize = float(sys.argv[3])

parsec = 3.1e18 #cm
distancetoobserver = 840 #parsecs
tmin = 0.0
tmax = 2.0*m.pi
pmin = 0.0
pmax = m.pi
energyvalues = []
binvalues = []

def angulardiameter(size, distance):
  return 2*m.asin(size/(2*distance))

angledelta = angulardiameter(observersize, distancetoobserver)/2.0
numanglebins = m.sqrt((len((inputfile.readline()).split()) - 1))
tbinsize = tmax/numanglebins
pbinsize = pmax/numanglebins)

tbinsfromcentre = int(round(angledelta/tbinsize))
pbinsfromcentre = int(round(angledelta/pbinsize))

tcentrebin = 0.0
temp1 = 0.0
for i in range(1, int(numanglebins)):
  temp1 += tbinsize
  if temp1 >= angle:
    tcentrebin = i
    break

pcentrebin = (numanglebins)/2.0

for line in inputfile:
  data = line.split()
  energyvalues.append(data[0])
  data.pop(0)

  temp = 0.0
  tindex = 1
  pindex = 1
  ite = 0
  while ite <= numanglebins**2:
    if (tindex <= (tcentrebin + tbinsfromcentre)) and (tindex >= (tcentrebin -
     tbinsfromcentre)):
      tindex += 1
      while pindex <= numanglebins:
        if (pindex <= (pcentrebin + pbinsfromcentre)) and (pindex >= (pcentrebin -
     pbinsfromcentre)):
          temp += float(data[ite])
          pindex += 1
          ite += 1
```

```
        else:
          pindex += 1
          ite += 1
      pindex = 1
    else:
      tindex += 1
      ite += int(numanglebins)
  binvalues.append(temp)

for i in range(len(energyvalues)):
  outputfile.write(str(energyvalues[i]))
  outputfile.write(" " + str(binvalues[i]) + "\n")

inputfile.close()
outputfile.close()
```

Listing B.4: Python spectral extraction script

The script takes in several parameters; the name of the input file, the cloud's azimuthal position (the clouds polar position is always taken to be $\frac{\pi}{2}$), the size of the observer and the output file name. The script then calculates a two required values: the size of the angular bins and the size of the observer in terms of angular diameter. It then calculates which azimuthal bin is the centre of the observer, this is where the cloud position comes into play, because what is actually happening in effect is not that the cloud is being rotated around the source, but that the observer is being rotated around the cloud. This just makes the code execution easier and is the same as a rotating cloud. The script then begins reading through each line of the main code output file. Each line is first split into a list (*data*), with each member of the list being a tab separated value in the output file (a different bin). The energy value of the bin is always the first value, so this is put into a different list and then removed from the main *data* list. The script then loops through the entirety of the *data* list, which is to say it loops through every angular bin for a particular energy. While doing this, it checks to see if any particular bin lies within the area of the observer. If it does, the photon count number from that bin is added to the total count for that energy and then appended in another list, *binvalues*. After looping through all energy bins in this way, the values are written to the output file as a spectrum.