Simulating a Broad-band Soft X-ray Polarimeter that Uses Changing Multilayer Mirrors

Jason Frost[1]

MIT Kavli Institute, Massachusetts Institute of Technology, Cambridge, MA 02139

jfrost@stanford.edu

Received ______________________; accepted ______________________

[1]Undergraduate student, Stanford University

1. Introduction

Soft X-rays from astrophysical sources have never before been viewed with a telescope that can detect polarization. Being able to observe polarization between approximately 0.2 and 0.8 keV would give us more information to help figure out which processes are responsible for the emission of these X-rays. While the physical lab setup for studying the polarimeter design is starting to provide results, it is also useful to be able to simulate potential polarimeter designs. This paper discusses the creation of a simulator for the polarimetry project that was built as part of the MARXS software, which is also being developed to design future X-ray observatories.

2. Design of the Additions to the Software

In this section, I discuss the features and classes that I wrote or had a significant role in. All of these classes rely on other pieces of code that were already part of MARXS. Also, please note that other parts of the software are very useful for simulating the polarimeter lab setup as well, such as the detector and grating classes or more generalized classes like the source, which is the parent of the lab sources.

2.1. Representing Polarization

In the MARXS raytracing software, each photon is tracked as it is generated by a source, interacts with various optical elements, and is recorded by a detector. Photons have a number of characteristics saved and recalculated throughout the program. These include position, direction, energy, and probability, among others. The polarimetry project required the MARXS software to handle polarization as well, so the representation of polarization had to be determined. Ultimately we needed to track the direction of the electric field vector to know the photons' polarizations.

One way to store the polarization of a photon was to save an angle between 0 and 2π. While easy to choose in a source module, this representation requires a known 0 direction. In most cases, this can be achieved by choosing one vector, say the unit vector along the y axis, $\hat{y}$. Then the vector corresponding to an angle 0 could be defined as

$$\hat{y} - proj_{\vec{v}}\hat{y}$$

because this is the vector perpendicular to the direction $\vec{v}$ and closest to $\hat{y}$.

The other option that was considered was representing the polarization as a vector. This requires no context, but is harder to choose initially so that the polarization vector is perpendicular to the direction. However, we ultimately went with this representation for most of the simulation because when the polarization is needed in calculations, having it in vector form is much more useful. This was important for the multilayer mirror, which will be discussed below. The angle representation is used only for choosing polarizations in a source and is quickly converted to a vector representation. The polarization vector is defined as one of the two unit vectors parallel to the electric field vector. This means it does not oscillate like the actual electric field. Thus, any polarization vector is exactly equivalent to a polarization that is the negative of the same vector.

2.2. The Lab Sources

As part of the simulation software, a lab source had to be added that behaved differently from an astrophysical source. This was essential for using the MARXS software to model the lab setup as well as a future launch. Two lab sources were written: a more efficient one called a far lab source and a slightly more accurate one simply denoted lab source. While both simulate a point source and can take any flux, polarization, and energy distribution as parameters, they calculate initial photon position and direction differently.

2.2.1. The Far Lab Source

The far lab source behaves like a point source and aperture, together. It emits photons from a rectangular region, and uses the location of the point source itself to find the direction of each photon. The direction is then just the vector from the point source location to the chosen starting point of the photon. This is a highly efficient way to produce photons in a particular direction, especially if the point source and aperture that are being modeled are somewhat spread apart. The only problem with this source is that it assumes a uniform distribution of photons within the rectangular aperture region. This is almost exactly correct for a sufficiently far point source, but can be a significant source of error if there is a large difference between the distance from the point source to the aperture's center and from the point source to the aperture's corner. If that is the case, the flux of photons through those places should be different because flux falls with r^2.

2.2.2. The More Accurate Lab Source

The second lab source is simpler, but much slower. It models a point source only and assigns every photon the same starting position. Direction is random so that photons are sent out in all directions (although there is an option to constrain the photons to one hemisphere.) This source is not feasible for simulations in which many photons would be lost and in which many photons need to be simulated.

2.3. The Baffle Plate

One of the simplest additions I made to the MARXS software was the baffle plate (or alternatively an aperture.) This class allows photons that intersect a given rectangle to pass through, and sets the probability of all other photons to zero.

2.4. The Multilayer Mirror

The most important component to the polarimetry lab simulation specifically is the multilayer mirror. This is where the photons are polarized, and it allows the polarimeter to distinguish between different bulk polarizations. This module was also the most difficult to program. The multilayer mirror must send incoming photons off in the correct direction; give those photons the correct probability of reflecting based on their intersection position, polarization, and energy; and recalculate the polarization of the photons.

2.4.1. Calculating the New Photon Direction

Recalculating the direction of the photons is fairly simple, as it only requires the normal vector to the plane of the mirror. Then the formula for the reflection is

$$\vec{v}_{new} = \vec{v} - 2\Big(\frac{\vec{v}\cdot\vec{n}}{\|\vec{n}\|^2}\Big)\vec{n} \tag{1}$$

where $\vec{n}$ is the normal vector to the mirror. However, the current code uses an alternative method in which it transforms the direction vectors to the mirror's local coordinates, multiplies the local x component by -1, and transforms the vectors back into global coordinates.

2.4.2. Calculating the New Polarization

Recalculating the polarization, while not difficult computationally, is a little more challenging to visualize. This was accomplished by dividing the polarization vector into two components: one perpendicular to the plane of incidence, s polarization, and the second parallel to the plane of incidence, p polarization (both must still be perpendicular to the photon direction.) Note that the plane of incidence is the plane determined by the span of the incoming and outgoing photon direction vectors. The next thing we had to do was determine where each polarization goes to after a reflection, when the wave is traveling in a new direction. In other words, we needed to determine how the electric field vectors in the s direction and in the p direction change immediately at the point of reflection.

According to *Physics of Light and Optics* (Peatross 2001), the s component should remain perpendicular to the plane of incidence, so it falls in the same line (see Figure 1.) However, according to the Fresnel coefficients, E_s changes sign and the vector $\vec{E}_s$ points in the exact opposite direction of where it pointed before the reflection and where it points in the figure. The p component of polarization must now be parallel to $\vec{k} \times \vec{E}_s$ where $\vec{k}$ is the photon's direction and $\vec{E}_s$ is the s component of the electric field (see Figure 1.) The Fresnel coefficients as derived in *Physics of Light and Optics* again show that E_p is negative after reflection, so that $\vec{E}_p$ points in the opposite direction as it does in the figure, for the reflected wave.

To figure out the new polarization, each photon is treated as a wave packet and is decomposed into a wave packet with s polarization and a wave packet with p polarization. This is done by taking the dot product of the polarization vector (which is normalized) and the unit vector in each of the s and p directions. The result is the cosines of the angles between the polarization vector and $\vec{E}_s$ and $\vec{E}_p$, $cos(\theta_s)$ and $cos(\theta_p)$ respectively. Then $\vec{E}_s$ and $\vec{E}_p$ are individually traced to where they point immediately after the reflection as described above, and the new $\vec{E}_s$ and $\vec{E}_p$ vectors are multiplied by $cos(\theta_s)$ and $cos(\theta_p)$, respectively, to get the vector projection of the new polarization vector in the new s and new p directions. These two projections are then added together to reconstruct the polarization vector after reflection.

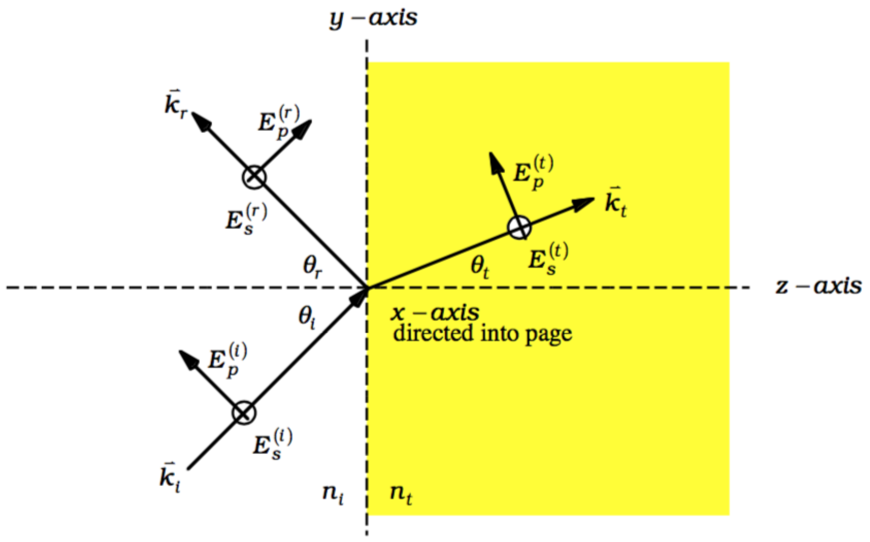

Fig. 1.— Conventions for the directions of the s and p polarizations, $\vec{E}_s$ and $\vec{E}_p$, from *Physics of Light and Optics.*

2.4.3. Calculating the Reflection Probability

The probability of reflection is calculated in two steps. The first factor that affects the reflection probability function is where a photon strikes the mirror. The second factor in reflection probability is the polarization.

At any position along the mirror, reflectivity is approximated by a Gaussian function of wavelength. The module takes as a parameter a table that provides the wavelength that is most reflective (the center of the Gaussian) and a measure of the width of the Gaussian (the full width half max) for every position on the mirror. This is used to create the reflection probability function, and the location where each photon strikes the mirror is plugged into the photon's corresponding probability function, based on its energy (or equivalently its corresponding wavelength.)

Polarization also affects the probability of a photon being reflected. This feature is what allows the multilayer mirror to polarize incoming light or reflect incoming polarized light at specific orientations only. s polarization, perpendicular to the plane of incidence, is the most reflective polarization. For this simulation, p polarization was assumed to have zero chance of reflection. Then, again treating photons as wave packets, the resulting amplitude of the reflected wave is equal to the projection of the incoming electric field E_0 onto the unit vector in the direction of s polarization.

$$E_{new} = E_0 \times cos(\theta_s) \tag{2}$$

The power carried by a classical electromagnetic wave (we are still modeling a wave packet) is proportional to the amplitude (maximum electric field strength) squared. To simulate all of this, the probability of the original photon reflecting is taken to be the probability of reflection for an s polarized photon times $cos^2(\theta_s)$, the square of the cosine of the angle between the polarization vector and s polarization.

$$P = P_s \times cos^2(\theta_s) \tag{3}$$

where P is probability of reflection. This model ensures that the expected total power of all reflected photons is equal to the total reflected power predicted by classical electromagnetism.

2.4.4. Potential Concerns with the Multilayer Mirror Model

In this section I list several concerns that occurred to me, but that I did not have time to thoroughly investigate. None of them seem particularly likely to be problematic, but I figured it would be helpful just to point out where the simulation's potential weaknesses might be.

1) Is it ok to treat photons as classical electromagnetic waves? They can be thought of as wave packets. However, we are taking the proportion of a diagonally polarized classical wave (having electric field composed of both p and s components) that should reflect to be the probability of a photon with that same diagonal polarization reflecting.

2) The Fresnel coefficients give reflectivity for p and s polarization, but we assume that p reflectivity is 0. See *Physics of Light and Optics* chapter 3 (specifically pages 48-51) for information about the Fresnel coefficients.

3) There could be a phaseshift in one of the polarization directions, but not the other (leading to circular polarization.) I don't think this is possible in our case, but I am not sure and do not know why. This is discussed in *Physics of Light and Optics* chapter 4.

3. Running the Simulation

There is still work to be done to make sure the simulation can completely replicate the lab setup and a potential flight mission. However, significant progress has been made, and the simulation can model the lab setup somewhat flexibly. One component that may need significant work going forward is the grating module. This might need to be adjusted or improved to make sure it accurately handles polarization and can model the various types of gratings that will be used in the lab.

Currently, the simulation is a python script that contains a number of functions that run the simulation with various setups (and can be given inputs to specify some distances and angles.) There are additional functions that draw several types of plots for visualizing data and that run certain simulations repeatedly with different inputs to produce a specific data set or to find the optimal positioning of a certain component. Two major steps in getting the simulation to run as it currently does were modeling the X-ray source in the lab and improving the efficiency to get significant data in a reasonable amount of time.

3.1. Modeling the X-ray Source

The X-ray source was already modeled by a number of IDL scripts, but to allow the simulation to model the lab source given any material, voltage, and current, large portions of these scripts were converted to python. The calculation involves two main steps. First, the bremsstrahlung spectrum (radiation produced by the acceleration of electrons in matter) must be calculated. Then the specific emission lines must be determined, modeled as Gaussians, and added to the bremsstrahlung spectrum. I will quickly summarize these steps because I did not write the code myself, but did need to understand it to combine it with the simulation.

Calculating the bremsstrahlung spectrum relied mostly on Kramers' law.

$$I(E) = \frac{K}{2\pi c\hbar}\Big(\frac{E_{max}}{E} - 1\Big) \tag{4}$$

where $E_{max} = qV = eV$ in J or $E_{max} = V$ in eV, and K is a constant that is proportional to the atomic number of the target element.

The energies of the emission lines are determined simply by the target material. The emission line energies are assigned in labx_target.py. After this, the emission line to bremsstrahlung spectrum ratio is calculated and multiplied by the bremsstrahlung flux at the emission line's energy to get the flux from the emission line.

The X-ray sources in the lab can currently be modeled with decent accuracy, but there is at least one feature that should be added. Right now, each emission line is assumed to be a delta function, but in labx_tubespec.py there is code that allows for "blurring" these lines.

This code is not currently used, but a quick edit should allow the user of the simulation to choose to widen the emission lines (the specific widths should be hardcoded in based on the target material.)

3.2. Improving Efficiency

When gathering data from the simulation, it is essential to deal with high volumes of photons, especially when they often reach detectors with extremely low probabilities. To run the simulation (on a laptop) in a reasonable amount of time, it was clear that the more efficient far lab source (see section 2.2.1) would be required. Using this source, the dimensions of the aperture were chosen so that very few of the generated photons would miss the detector. Another way I maximized the amount of data collected was to choose whether or not the photons reached the detector at the very end of the simulation, using the composite probabilities, so that photons could be selected multiple times. This strategy is justified because the simulation does track a very large number of photons that reach the detector with some small probability. Thus, the simulated photon distribution sample should be representative of the actual photon distribution. The reason only a small number of photons are chosen to actually reach the detector is that the probabilities are very small, so it is ok to choose photons multiple times.

4. Results and Conclusions: Just the Beginning

Preliminary results from the simulation have agreed with expectations. In a source-mirror-mirror-detector setup similar to the lab, rotating the source and first mirror together for an entire rotation yielded the most detected photon counts (after selecting photons based on their probabilities) when the source-first mirror axis was parallel to the second mirror-detector axis. The least photons were detected when the axes were perpendicular, and plotting photon counts every 10° revealed a sinusoidal wave, as expected.

An earlier interesting result from a simulation with just a single mirror and a monochromatic source showed a diagonal line of photons detected. This showed that the photons were reflecting off of the mirror correctly, and that photons of any given energy were only reflected by a thin band of the mirror, as desired.

The simulation is still a work in progress, but it is already useful for performing some simpler experiments. As the software continues to be developed, this should be a useful tool in the design and testing of the lab setup and the eventual flight mission.

Thank you to Herman Marshall and Moritz Guenther for all of your help throughout the project!