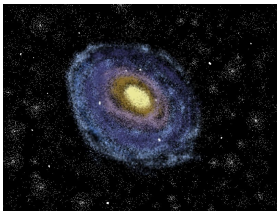

Team Andromeda

Version 2

Requirements Specification

December 12, 2019

Accepted as baseline requirements for the project:

For the client: _____ _____ *For the team:* _____ _____
Signature *Date* *Signature* *Date*

_____ _____
Signature *Date*

Clients - Dr. Audrey Thirouin and Dr. Will Grundy
Mentor - Isaac Shaffer
Members - Matthew Amato-Yarbrough, Batai Finley, Bradley Kukuk, John Jacobelli, and Jessica Smith

Table of Contents

1. Introduction 1

2. Problem Statement 2

3. Solution Vision 3

4. Project Requirements 6

5. Potential Risks 16

6. Project Plan 18

7. Conclusion 19

1. Introduction

Since the time of Galileo, humans have been relentlessly studying the universe. Currently, billions of dollars are spent every year on sending probes to other planets and small bodies in an attempt to understand what lies in the cosmos. This continuous research has driven technological development in areas not directly related to space. For example, home insulation, baby formula, and portable computers are a few of the side results discovered through space exploration. Other valuable data and theories have been derived from space exploration as well, including information such as early planet formation and evolution. Many observatories like Lowell are gathering information daily to further our comprehension of areas in space such as the Kuiper Belt and how small bodies form and interact with each other.

Our clients Dr. Audrey Thirouin and Dr. Will Grundy work for Lowell Observatory. Dr. Thirouin is a research scientist interested in the characteristics of small bodies within the Solar System, while Dr. Grundy is an astronomer that researches Kuiper Belt objects. Kuiper Belt objects are a region of leftover bodies from the solar system's early history, making them valuable for observing conditions similar to that of early planet formation. Together, Dr. Thirouin and Dr. Grundy focus on collecting and analyzing data about small bodies that lay far from Earth and are hard to directly observe. Currently, they are working on modeling binary systems in the Kuiper Belt and need to use techniques that make the most use of the data available to achieve this. For distant objects, only a pinpoint light source is observable. This can be used to determine the brightness of the object over time. These luminosity recordings can be combined to form a light curve, which is a graph of brightness values over a given set of time.

Light curves can be used to infer rotational and physical properties of small bodies. For instance, an asteroid that is non-spherical will reflect more light when a larger portion of its surface is facing an observer. This is because more light from the Sun is being reflected to said observer. Since the object is reflecting more light at certain points in its rotation, the brightness will differ depending on when it is observed in its rotation. The various brightness values can then be graphed, generating the light curve. Information such as the period and amplitude of the curve, rotational period, and rough properties can then be found based on graph data. A broad number of other characteristics can be estimated using light curves by an adept astronomer.

Our clients wish to use light curves to better understand binary systems. A binary system is composed of two objects that orbit a common barycenter. A barycenter is a point in space that represents the center of mass of two or more orbiting bodies. The gravity from both bodies affects the other and can cause unique situations, such as tidal locking or precession. Binary systems introduce new challenges while also offering opportunities for light curve modeling.

Dr. Thirouin and Dr. Grundy can model binary systems with spheres and faceted objects using software developed in a previous iteration of the project. Our clients need upgraded software to improve the application by including triaxial ellipsoids, a Markov chain Monte Carlo (MCMC) algorithm, a graphical user interface (GUI), and a video generator. The solution for this iteration of the project will allow for more accurate modeling of binary systems and the generation of estimated parameters of binary systems. This increase in accuracy and ease-of-use will advance Lowell's research efforts to analyze binary systems in the Kuiper Belt.

2. Problem Statement

2.1 Current Workflow

Our clients use the solution to model binary systems by inputting over fifty parameters into an IDL command line. From here, they choose to either use spheres or faceted objects to represent the binary system they are simulating. After this is done, the Forward Model is executed and returns a simulated light curve of the system. Depending on the quality of the modeled light curve, they will either adjust parameter input and make a new light curve or keep the produced predicted light curve.

2.2 Current Workflow Issues

Currently, the clients are facing a host of issues that are disruptive to their workflow and limit the capabilities with which they can model binary systems. They need the addition of new modules to strengthen the software solution they currently have implemented. These modules will address the client's following issues:

- Inability to represent a triaxial ellipsoid-shaped body
- Parameter input into the command line is inefficient

- Predicted parameter estimates are done by hand
- No option to produce videos from rendered images

The current solution can only use spheres or faceted objects to represent binary systems, which in turn limits our clients to two choices. They can either choose to render binary systems quickly with low accuracy using spheres, or more accurately with longer render times using faceted objects. Additionally, command line input makes for a needlessly arduous and time-consuming data entry process.

Furthermore, the workflow currently lacks an efficient means to determine the best fitting parameters for an observed light curve. To determine these parameters, our clients need to manually perform the operations that a Hamiltonian Monte Carlo algorithm is capable of. This again further impedes the workflow of our clients, adding a significant amount of overhead to what could otherwise be an automated process. Lastly, without a means to take rendered images and compile them into a video format, our clients lack the ability to visualize the movement of the binary systems being simulated without manually doing so.

3. Solution Vision

3.1 Overview

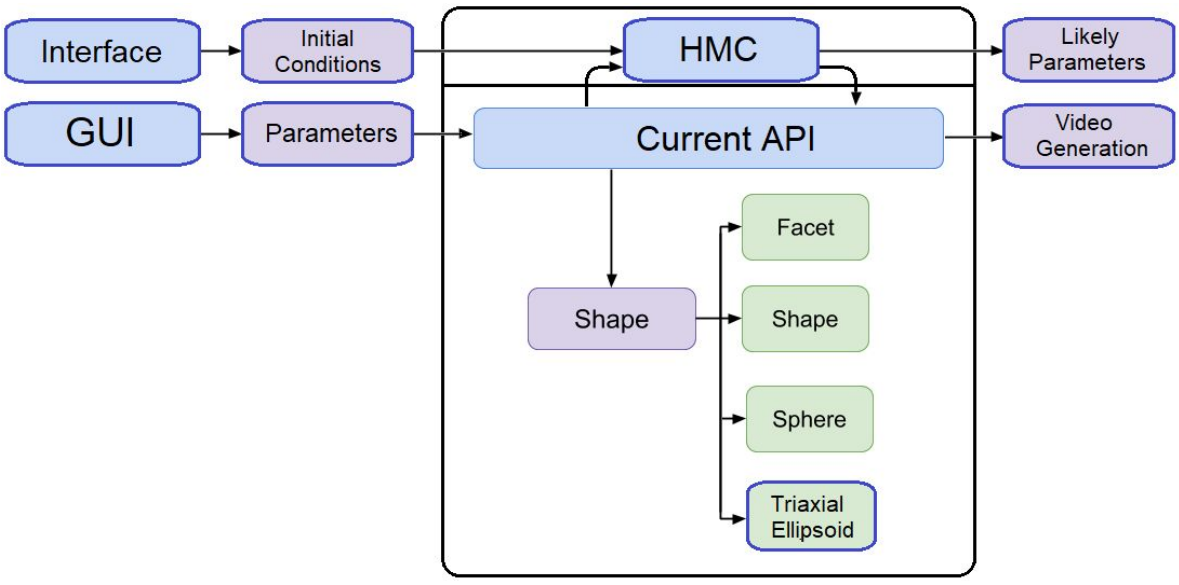

Figure 1: Envisioned Forward Model used in conjunction with the HMC API

Our solution will address our clients' problems by using the current API, frameworks, and math libraries to improve performance. A framework will be used to provide a GUI to facilitate the entry of parameters for the Forward Model. Additionally, the GUI will allow for the option to compile rendered images into a video format. The solution will have a Shape subclass for the implementation of triaxial ellipsoids. Lastly, the solution will use an HMC API to handle the implementation of the HMC algorithm.

Figure 1 above is a simplified, modified version of the current Forward Model with our envisioned improvements. It only shows existing modules relevant to our work this year. The portions of the Forward Model that have a blue outline are what our team plans to implement, with everything else being implemented in the current software.

The API will be well documented to allow users to maintain and update the code base. The design of the API will be modular to allow for new functionality to be added in the future.

3.2 Solution Statement

Due to the requirements given and the existing code base, the software that will be developed must be compatible with the existing code base. The existing code base was developed using C/C++. With that in mind the software will be built using C/C++ and Python.
For the GUI implementation for the Forward Model, the programming language we will be using is Python due to its compatibility with C/C++. The framework that will be used to develop the GUI will be Kivy, a Python framework that is used to create cross platform compatible software. This will allow the user to use this software anywhere and at any time.

For the acceleration of the Forward Model we will be creating a Triaxial Ellipsoid class to the existing API using C/C++. The Triaxial Ellipsoid class will be dependant on the existing API's library and will use all of the same libraries.

The API will be well documented to allow users to maintain and update the code base. The design of the API will be modular to allow for new functionality to be added in the future.

3.3 Solution Steps

Graphical User Interface

The solution for GUI implementation is to wrap the existing API in a GUI. The clients want the interface to take in parameters for the Forward Model so that it can call the Forward model, and generate a light curve to be displayed to the screen. The GUI that will be developed will need to be able to do the following:

1. Run the Forward Model
2. Take in input parameters and accuracy settings
3. Display the generated light curve
4. Allow different Forward Model settings

Model Acceleration

The solution for Forward Model Acceleration is to create a Triaxial Ellipsoids class. The clients want a Triaxial Ellipsoid class that can be used to calculate and render objects more accurately so that the Forward Model can more accurately model what the observed object looks like. The Triaxial Ellipsoids implementation that will be developed will need to be able to do the following:

1. Evaluate orientation from spin state
2. Accurately calculate orientation for ellipsoids

Hamiltonian Monte Carlo Algorithm

The solution for the HMC algorithm implementation is to wrap the Forward Model with a HMC algorithm. The algorithm will be used to give a likely set of parameters based on the observed data set. The Hamiltonian Monte Carlo algorithm that will be developed will need to be able to do the following:

1. Produce a range of likely parameters for Forward Model
2. Will take in initial conditions and observed data

4. Project Requirements

4.1 Functional Requirements

There are three major use cases that must be satisfied for this project, which are the implementation of an ellipsoid object, a GUI that passes parameters into the current API, and the addition of an HMC algorithm to produce likely parameters for an observed light curve.

Our improvements to the Forward Model involves 3 major steps:

1. Implementing the Hamiltonian Monte Carlo algorithm
2. Rendering and calculating the light curves of triaxial ellipsoids
3. Improving user input and model output through the addition of a GUI and video generation.

These extensions will be used to improve the current Forward Model by allowing for a greater ease of use for our clients.

FR1. Simulate Triaxial Ellipsoids

The software must be capable of performing calculations for the Forward Model using triaxial ellipsoids. It must also be able to display a triaxial ellipsoid object when calling the Forward Model. This output will be generated through the use of the already implemented ray tracing feature. Additionally, this output must be compatible with the modifications that the user may request of other implemented objects by way of input parameters. These adjustments to the output include the ability to adjust the resolution or pixel count of the image. The requirements of simulation and calculations of triaxial ellipsoids can be broken down into four sections:

1. Calculate light curve using triaxial ellipsoids

The current iteration of the API takes in a given amount of parameters to calculate the light curve for a system, including:

- Ephemeris table
- Keplerian orbital elements
- Time value(s)

- Object shape parameters
- Spin states
- Hapke parameters
- Optional accuracy setting

The software must be capable of performing calculations for the Forward Model using triaxial ellipsoids. All of these parameters work in conjunction to calculate the forward model, and provide a light curve of a system. The triaxial ellipsoid object will need to be compatible with all the above parameters which the API utilizes in order to calculate a light curve.

2. Render triaxial ellipsoids as objects

Rendering images of the system during the calculation of a light curve is also possible. The software must be able to render a triaxial ellipsoid object when the user requests so when calling the Forward Model. As the ellipsoid object will be used by the Forward Model, its model output will be generated through the use of the already implemented ray tracing feature.

The current program simulates objects, meaning this will also be required of triaxial ellipsoids. Rendering the triaxial ellipsoids will allow users to visualize how the objects are interacting with each other. Ray tracing is the method currently used to render objects in the program. Spheres use a hit-or-miss calculation to determine whether a ray hits, finds the angle of reflectance of that ray, uses those angles with the hapke bidirectional reflectance model, and then gets that return and saves it as a relative luminosity. This is then later corrected for the image. Triaxial ellipsoids will need to be compatible with this process to be renderable.

3. Allow for manipulation of dimensions

Triaxial ellipsoids describe a large group of shapes. Spheres that have been shortened or lengthened along their x, y, and/or z axis in a 3D space are considered to be a triaxial ellipsoid. Being able to modify the value of triaxial ellipsoid's axes is key to implementing these shapes and allowing the user to gain the most from them. These parameters must be clear to users so that they know how to change the shape of the ellipsoid.

As the ellipsoid object will be used by the Forward Model, its model output will be generated through the use of the already implemented ray tracing feature.

4. Rotate object

The sphere class that is implemented in the current program does not have to consider rotation, as a sphere is uniform throughout. Triaxial ellipsoids are not uniform throughout, and therefore must consider the possible fluctuations due to rotation. An orientation function must be implemented for the shape to account for these fluctuations, and will allow for the model to accurately represent the shape.

The rotation will be the most difficult part of the shape to implement due to the math that is needed to calculate it. This will be accomplished through matrix multiplication that will update the ellipsoid object depending on what orientation is required.

FR2. Produce Best Fit Parameters Using HMC

A key requirement of our project is the implementation of an HMC algorithm. This algorithm will serve the purpose of finding the best set of parameters to match the observed light curve to the modelized one. Additionally, the algorithm will show how well our clients have pinned down the values that they are interested in, and visualize this cloud of solutions using a model, such as a Corner Plot. Lastly, the algorithm will save the solutions produced to an external file. The requirements of the algorithm are explored in further detail in the following sections.

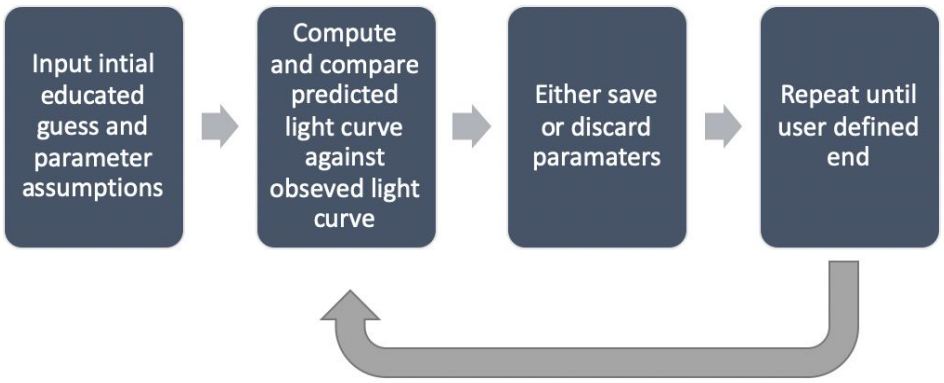

Figure 2: Steps of the HMC Algorithm

1. Produce likely parameters that fit the observed data

The main functionality of the HMC algorithm will be to produce estimated parameters that best fit the observed light curves. The algorithm accomplishes this by inputting an initial educated guess and parameter assumptions into the Forward Model. This in turn produces a predicted light curve that is then compared to the observed light curve. If the comparison falls within a certain range of acceptability, then the parameters used to create the predicted light curve are saved as a possible set of solutions. Otherwise, the parameters are discarded and a new educated guess is made. This process is repeated, with each set of tested parameters being determined by the previous solution. This process continues until a user defined end. The steps of the algorithm are laid out in Figure 2 above.

2. Display the range of solutions using a model

Using the model produced by the HMC algorithm, our clients will be able to observe the relationship between the parameters that were considered by the algorithm. Our clients suggested using a Corner Plot to display the data produced by the algorithm. It is a method to visualize multidimensional samples, such as those produced by an HMC algorithm, using a scatterplot matrix.

3. Saving the solutions to an external file

Our clients would also like for the HMC algorithm to produce a file, such as a .csv file, that would contain the raw data produced by the HMC algorithm. If we imagine the layout of the file as a table, the number of columns would be equal to the number of variables that were sampled from, and the lines would be equal to the number of samples that occured. This information can be used in testing outside of the algorithm or reused as input for subsequent algorithm runs.

4. Inputting data from the previous run into the HMC algorithm

Moreover, our clients asked us to include the functionality of taking the output of one HMC algorithm run and inputting a section of this output into a subsequent algorithm run. For example, after running the HMC algorithm with the initial parameter assumptions, the subsequent run would use the median associated with one of these parameter assumptions as its starting reference for that particular parameter. This median can be pulled from the external file outlined in the previous section. In this way, the algorithm would have a head start when it comes to calculating the best value for that specific parameter.

5. Stretch Goal: Annealing functionality for the HMC algorithm

Additionally, our clients want the inclusion of annealing for the HMC algorithm. Annealing would limit the ability for the HMC algorithm to traverse the parameter space so that the most likely estimates for each parameter are determined in a more efficient manner. This process of annealing would occur after the burn-in period. This will be a stretch goal for the HMC section of our project.

FR3. Implementation of a GUI

A key requirement that we have is the implementation of a new GUI for the forward model. This GUI will make the Forward Model more user friendly and allow the user to enter parameters from the interface rather than entering them at the command line. Additionally it will allow the user to see the predicted light curve directly on the screen where the user will be able to compare the predicted light curve to the observed light curve.

1. Input to Forward Model

The GUI must allow the user to enter in variables from the interface. There will be text boxes that will take in a specific type of input, and if the input type is invalid, the GUI will prompt the user to put in an appropriate input. Once all variables are entered, there will be a button that will allow the user to run the Forward Model with the input parameters.

Due to the large amount of parameters to be input for the Forward Model, there will be a scroll wheel that will allow the user to navigate through the parameter list faster.

2. Output to Forward Model

Once the software has acceptable parameters, it will enter those parameters into the Forward Model and generate a light curve that will be displayed to the screen. Once the light curve is generated to the GUI the user will be able to save the output as a singular image locally on their device.

There will be a tab at the top of the output panel that will allow the user to switch between the generated light curve and the rendered images from the Forward Model.

3. Settings

The GUI will start with a standard default for setting configurations. For the settings tab, the user will be able to select a settings and depending on what that setting should do, it will dictate variable input on the GUI for the variables that will not be used, or set those variables to a predetermined constant. There will be a large range of settings that will be implemented allowing the user more flexibility in variable input while also constraining those variables that will not be used.

FR4. Generate Video From Forward Model

Our clients want to make the process of generating videos from their images more streamlined. Currently, the program will create images of the model, but does not produce a video. Our clients would

like to generate a video of the images that the model would produce. The video generation is broken down into three main requirements below.

1. Output File

When the video generation software is run, it will take in a collection of images and condense them down to a .MP4 file. The video generation software will be accessible from the Forward Model GUI, and the file location will be put into a text box. From there, the images will be converted to a .MP4 file at the press of the “Compile Button”.

2. Camera Control

We also wish to give the user some control over the pointing of the “camera” in video production. Currently, images are generated while centered on a specified body. This added control would allow the user to choose which body should be the center of the video, and how wide the field of view should be. The user will also be able to control the start,end time, and the time step of the produced video.

3. Image Collection

As the API renders objects, it will save images of the binary system model into a folder. The images will be a visual representation of the binary system as the individual objects complete their orbit. These images will be rendered from the Forward Model while it is running, and then later used by the video generator to create a video.

FR5. User Interface for HMC API

1. Input into the HMC API

Our clients asked us to include the functionality of a command line interface for the HMC API, in which parameters and their subsequent values can be passed into the API via a command line. Moreover, they asked that the method in which the parameters would be passed into the command line interface could be in the form of typing the command line by hand or by inputting a file that can be parsed for parameters.

In regards to writing the command line by hand, we would include a separate GUI for the HMC API interface that would create the command line for our clients. This would expedite the process of constructing the command line, as well as ensure that the command line is legal, or is constructed in the correct fashion.

As for inputting the parameters via a file, we would include the ability for the command line interface to take in a file that would contain the necessary parameters in order to run the API. This would be appropriate for tests where the values being input into the API are similar to previous tests, with only a few values being changed.

2. Stretch Goal: HMC API interface within the Forward Model GUI
A stretch goal for the HMC API interface would be to include the option within the GUI of the Forward Model that would send the values being tested, directly to the HMC API. This could possibly be in the form of a separate tab in the GUI of the Forward Model for the HMC API specifically, where the values input in the Forward Model tab could be "carried" over to the fields in the HMC API tab, and vice versa. This would eliminate the need to use the command line interface in order to run the HMC API whenever our clients are also running the Forward Model.

4.2 Performance Requirements

Currently our clients' solution has room for growth in order to help streamline the overall process. Through our additions, our clients will have an easier work flow for the model. We still must be conscious of the user, especially with requirements such as usability, and understand how the program is expected to perform. Though many of our implementations' performance will be up to the users, we have broken down and detailed our core requirements below.

PR1. Runtime of HMC Algorithm

The runtime of the HMC Algorithm varies depending on how long the user wishes to run the algorithm along with the type of shape

they use for the Forward Model. More steps result in a longer runtime which can lessen our clients' productivity. Without a user being able to choose when the algorithm ends, it could run an indefinite amount of time. A user defined end will be used for this algorithm to ensure that the process can be halted when the user believes it has collected enough data in regard to the variables they are interested in.

While a minimum runtime cannot be determined because of the user defined end, the user must be aware of what they are looking for and consider this when utilizing HMC. Still considering the defined end, there is also no defined accuracy of the algorithm. The HMC Algorithm will be utilized as a scientific tool, thus, the output is largely determined by which parameters and settings the user chooses as input for the algorithm.

PR2. Graphical User Interface (GUI)

The GUI that is planned to be implemented to help simplify the process for our clients will be simple to use. Though we are in continuous communication with our clients to ensure that the design of the GUI is to their specification, other users must be considered. The design will be simplistic and not require much learning from the user. Input for each parameter will be labeled so that the user knows where each parameter belongs. As numerous parameters are needed to use Forward Model, users will be provided with an explanation of how each parameter influences the Forward Model within the provided documentation. This will be provided in the user guide so that there will be an easily accessible reference.

We can consider the video generator in this section as well, as the user must use the GUI to determine whether a video should be output. This will be a simple button that is fairly intuitive.

PR3. General Usability and Readability

The functional requirements implemented within our solution will be supported with extensive documentation. Explanations for all the

functions we develop, and their parameters will be provided. Additionally, tutorials of how to properly utilize the GUI, generate videos within the GUI, and usage of the HMC algorithm will be given to ensure as low a learning curve for our clients as possible. The documentation we supply will enable other users beyond our clients with the information necessary to utilize this application.

4.3 Environmental Requirements

ER1. Cross-Platform Compatibility

The solution we implement needs to be able to compile and run on Linux, Windows and Mac. This is an environmental requirement met by the team that worked on the previous iteration of this project. Our clients asked that we keep this functionality with the solution we implement in order to maintain code usability and shareability.

4.4 Conclusion

The solution that we plan to implement must satisfy the aforementioned functional, performance and environmental requirements. These functional requirements include simulating objects in binary systems as triaxial ellipsoids, and computing the best fit parameters of an observed light curve using an HMC algorithm. Additionally, the solution will need to contain a constructed GUI for the Forward Model, as well as the ability to create a video from rendered images produced by the Forward Model.

Within meeting these functional requirements, our solution must also consider several performance and environmental requirements. This includes requirements such as the runtime of the HMC algorithm, the knowledge needed to utilize the GUI, and the general usability and readability of the code we implement. Lastly, the solution will consider the requirements of cross-platform compatibility and the limitations of computers that will be used by our clients.

5. Potential Risks

It is important to clarify any risks that could arise during the implementation process. While the risks we identify are subject to change, it is important that they will be addressed with a plan for mitigation. After combing through our project requirements, we found three specific risks that could impact the project throughout the year. The first risk is scope expansion, which regards the fact that our project will have future iterations. The second risk references the learning curve derived from the HMC algorithm. Lastly, the third risk addresses the rotation function of the triaxial ellipsoid class.

5.1 Scope Expansion

Our clients envision this project to be revisited in future capstone sessions. As per the original project requirements, Audrey and Will would like to add on a way to represent n-bodied systems and optimize the program for running on GPUs. Our clients would also like to see rings and brightness distribution added as a functionality in the future. To accommodate our clients' vision for the future of this project, we need a mitigation plan to counteract any issues with future integration.

Mitigation

Paired Planet Technologies, the team that created this program last year, implemented modular design concepts to uphold future integrations. To mitigate the future risk of scope expansion, we will hold ourselves to those same modular design standards.

5.2 Hamiltonian Monte Carlo Learning Curve

The Hamiltonian Monte Carlo algorithm is the most important part of our project. After much deliberation among our team, our mentor, and our clients, it is apparent that it is also the most challenging factor. Our team does not have a strong background in Bayesian statistics, and our clients do not have formal education on the inner workings of the Hamiltonian Monte Carlo algorithm. Since the HMC appears to have a steep learning curve, we need a mitigation strategy to counteract the likelihood for problems to arise due to misunderstanding.

Mitigation

To counteract the steep learning curve stemming from HMC, we need a vast support system. This support system includes our mentor Isaac Shaffer, who has a Master of Science degree in Statistics. In addition, we will be reaching out to the Statistics department of Northern Arizona University. It is important that we keep our clients updated on any knowledge gained from this support system. To sustain a solid understanding for our clients, we will update them during our client meetings.

5.3 Triaxial Ellipsoid Rotation

As previously discussed in the Scope Expansion section pertaining to Shape Class, the triaxial ellipsoid class is mostly finished. Paired Planet Technologies had added the Ellipsoid class to another branch of the project. However, they were unable to solve the equation for the spin state of a triaxial ellipsoid within the given time. Since the radii of triaxial ellipsoids may vary greatly, it is challenging but important to have a rotation factor to get characteristics from all angles.

Mitigation

The orient function of the triaxial ellipsoid class was made previously to perform the rotation. However, Paired Planet Technologies did not have sufficient time to get it functioning. Since the shape class for triaxial ellipsoid was already created and we only need to fix the orient function, there is no mitigation plan needed. There will be a slight learning curve to the math behind the rotation however, and therefore we will update for a mitigation plan if needed.

5.4 Risk Analysis

In order to present these values in a simple layout, we created a risk analysis table. The risk analysis table has a tab that holds the challenges that we face. The severity tab ranks (1-10) how important that challenge is to our current project. The likelihood tab shows a percentage (1% - 99%) for the challenge, representing how likely it is to cause problems. The risk

value is calculated by multiplying the severity by the likelihood. For risk values above a value of 3, a mitigation strategy will be enacted.

Risk Analysis Table				
Challenge	Severity (1 - 10)	Likelihood (1%-99%)	Risk Value	Mitigation
Scope Expansion	5	99%	4.95	Modular design
HMC Learning Curve	9	99%	8.91	Heavy communication
Triaxial Ellipsoid Rotation	8	10%	.8	Not a problem

Table 3: Project risk factors briefly outlined

After considering the severity of these risks and how likely they are to cause issues, scope expansion and the learning curve associated with HMC needs to be handled. Since this project will be revisited in future capstone projects, we will deploy modular design tactics to uphold succeeding expansions. We will also heavily research HMC and build a knowledgeable support system to counteract the learning curve associated with it. We conclude that fixing the triaxial ellipsoid rotation function is not a problem as of right now. Team Andromeda recognizes that the risk values related to the rotation function may change and will be updating our requirements specification as adjustments are needed.

6. Project Plan

To ensure the success of our project, we reviewed our short term and long term goals. Our current work regards the beginning tasks for building the alpha prototype. Our future work peers into the second semester. The second semester will be a crucial period of time where we will be finishing the alpha prototype and refactoring code. These tasks can be viewed in Figure 4, which is located at the end of the document.

6.1 Current Work

There are three critical aspects of our project that will be implemented in December. These aspects include the addition of triaxial ellipsoids, researching the HMC algorithm, and designing the GUI. We have added the triaxial ellipsoid class to the master branch of the project thus far. This addition led us to make a unit test that outputs a PNG file of the ellipsoid. Moving forward, we will be working on the 2D rotation of the triaxial ellipsoid before diving into 3D rotation next semester. For the HMC, we have attempted to replicate a tutorial of the algorithm in Pystan so that we can have a better understanding of the task at hand. We are also keeping in touch with our clients and our mentor to uphold a good level of understanding. As for the GUI, we are working with our clients to find a design that is efficient and effective for the large amount of variables involved in the parameter estimation process.

6.2 Future Work

Looking into the second semester, there are four portions of our project we will need to implement. Since the triaxial ellipsoid class is mostly implemented, we will only need to correct the orient function so that spin state can be utilized. We will also consider merging the Sphere class into the Ellipsoid class. Our client would like us to keep a copy of the Sphere class separate until we are able to confirm that their runtimes are roughly equivalent. We will also need to fully integrate the HMC algorithm, the GUI, and the video generator.

Since Paired Planet Technologies previously designed this project modularly, we will maintain that design throughout our integration. Once we have finished the alpha prototype in March, we will spend the rest of the semester optimizing our solution.

7. Conclusion

Space is an exciting, mysterious field. Despite thousands of years of research, humankind has only begun to scratch the surface of understanding the universe. The New Horizons mission, launched in 2006, started its maiden voyage to the Kuiper Belt in pursuit of gathering characteristics of Pluto and

other Kuiper Belt objects. To this day, there are numerous objects floating in the Kuiper Belt that we have yet to explore. Up to 30% of those asteroids are considered to have at least one secondary in their orbit. Studying these binary gives the power to unlock answers for how they are formed, how their orbits work, and how they fit in to the solar system.

Since space voyages are time consuming and costly, telescopic observations from earth are the most consistent way to form hypotheses of binary systems. Scientists can prepare observed light curves but cannot easily confirm the correctness of the characteristics formed from it. Our project was purposed to help Will Grundy and Audrey Thouirin at Lowell Observatory solidify these attributes.

The Hamiltonian Monte Carlo algorithm we are implementing is the key to producing genuine binary asteroid characteristics. The addition of the triaxial ellipsoid shape will accelerate render speeds for visualization without a high cost of accuracy lost. The HMC algorithm will allow for computed estimations of parameters for observed light curves, allowing for easier gathering of characteristic information for a binary system. The GUI will streamline the data input process for our clients, easing their use of the system. Finally, the video generator will compile a movie for our clients to use at will. We understand that the HMC algorithm is a daunting but important task, and therefore we are deeply researching the problem.

Team Andromeda is extremely excited to continue working with Dr. Audrey Thirouin and Dr. Will Grundy. Their observations at Lowell Observatory are yielding continuous advancements in the understanding of our solar system. We are delighted to be contributing to the progression of space knowledge and are ready for the challenge.

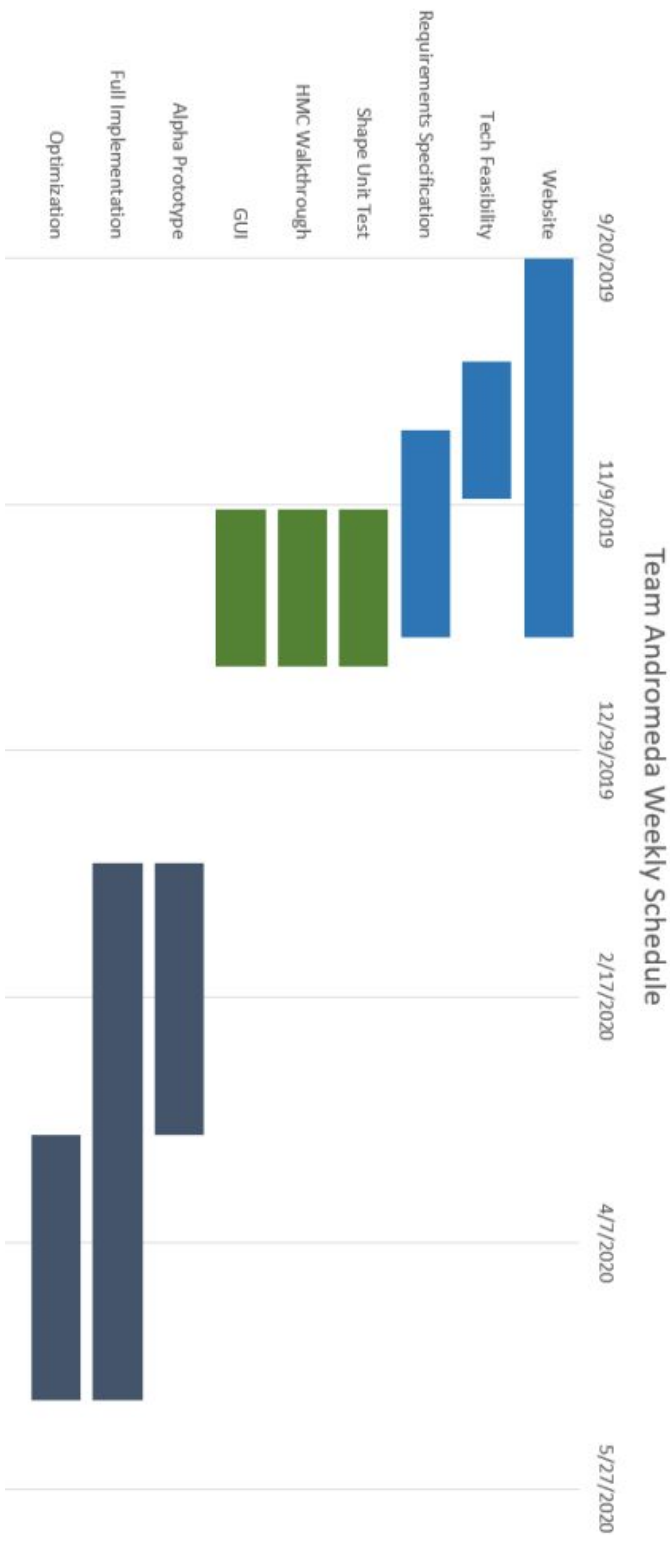

Figure 4: Gantt chart outlining weekly schedule