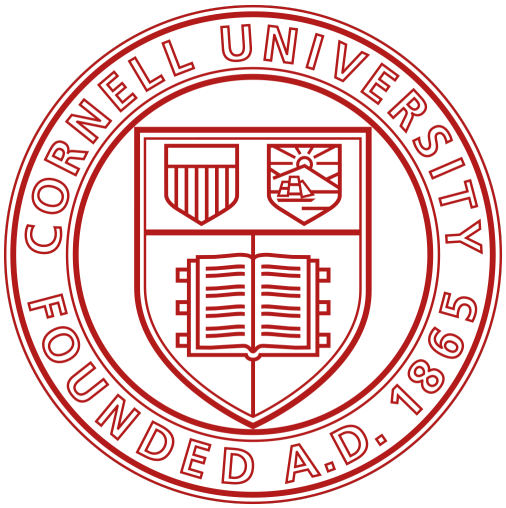


Cornell University

Master of Engineering Project Report

EXOSIMS Software Development: Building an Integration Time Calculator for the WFIRST Coronagraphic Instrument

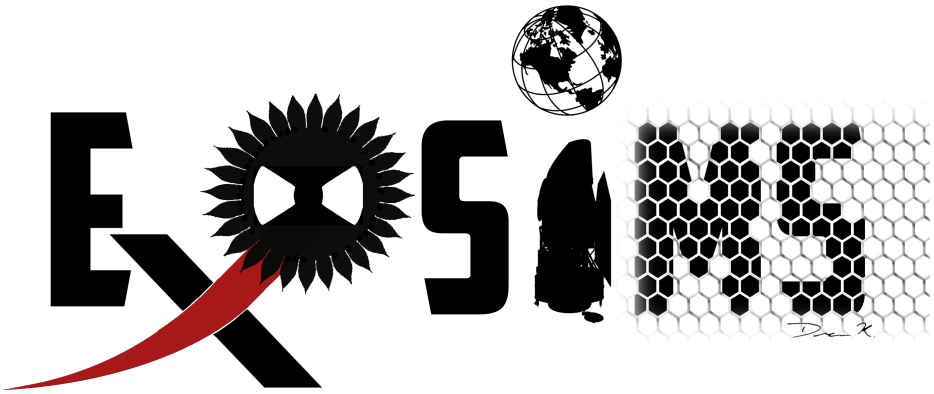
Jeremy Turner

Supervisor:
Dr. Dmitry Savransky

Spring 2019

Abstract

One of the science objectives of the upcoming NASA Wide Field Infrared Survey Telescope (WFIRST) is to directly observe exoplanets via high contrast coronagraphy. The planned 2.4 m telescope will be designed to detect objects on the order of a billion times dimmer than their host stars via imaging and spectroscopy. Given the weak signals from exoplanet targets, it is necessary to construct a highly descriptive model of all noise sources and throughput effects in the telescope and coronagraph instrument. The Exoplanet Open-Source Imaging Mission Simulator (EXOSIMS) is designed to serve as a comprehensive model of this system. EXOSIMS is an object-oriented software framework implemented in Python through which various stand-alone modules can be configured to simulate exoplanet observations with customizable mission, observatory, and instrument settings. This will allow scientists and engineers to investigate the system's performance and to estimate science yields. Critical to designing an observation is the integration time, the amount of time which much be spent observing a target to reach a desired signal-to-noise ratio (SNR). My work on EXOSIMS has been to implement a method to calculate integration times based on the rates of detected photons due to the exoplanet, its host star, and noise sources. This calculation is modeled on the work of Dr. Bijan Nemati at the Jet Propulsion Laboratory and currently at the University of Alabama in Huntsville. This work required me to cohesively integrate various calculations and equations as Python code into the EXOSIMS framework. Additionally, I produced a script which allows my code to calculate integration times for existing exoplanet orbital data. This report describes the underlying mechanics of coronagraphic observation, how they are modeled, and how the resulting model is implemented in software.



Contents

1 Introduction

The Wide-Field Infrared Survey Telescope (WFIRST) will be a 2.4 m space telescope expected to launch in the mid-2020s.* The mission will attempt to answer the questions: "How does the universe work?", "How did we get here?", and "Are we alone?", as laid out by the NASA 2014 Science Plan [1]. The last of these questions is addressed by the quickly growing field of exoplanet research within astronomy. Thanks largely to NASA's Kepler mission, the number of confirmed exoplanets is now measured in the thousands. WFIRST will contain a coronagraph, an instrument used to block out the light of a star to allow direct imaging of exoplanets. This will allow the telescope to observe planets down to 10^9 time dimmer than their host stars, such as sub-Jupiter size planets orbiting within a few AU [2]. Because WFIRST will be measuring dim sources, it is important to have a good estimate of how long the telescope must spend looking at a source, or integration time. In service of this, the Exoplanet Open-Source Imaging Mission Simulator (EXOSIMS) is being developed as a complex, accurate model of the actual coronagraphic instrument (CGI). This object-oriented software written in Python will allow users to scrupulously model an exoplanet observation to estimate science yields and necessary integration times.

1.1 Exoplanet Detection

The first unambiguous detection of a planet orbiting another star occurred in 1992 [3]. In the ensuing two decades, hundreds more were discovered. At present, there are over 4,000 confirmed exoplanets. This has been made possible by an array of tools and methods developed by astronomers, making full use of the electromagnetic spectrum and our current understanding of physics. One of the earliest and most robust detection methods uses radial velocity measurements of stars. This relies on the Doppler effect: when a light source is moving relative to an observer, the wavelength of its light lengthened (moving away) or shortened (moving towards). When a planet orbits a star, the star also orbits the planet.† Thus, over the course of an orbit, a star can be seen to move back and forth via its Doppler shift. A related method involves pulsar timing. Pulsars are rapidly spinning neutron stars which emit a beam of light observed each rotation. Their rotations are so stable that subtle variations due to the pull of exoplanets can be measured. This was the method used in the first confirmed extra-solar planet discovery [3]. A star's movement due to an exoplanet is also detectable by astrometry, the precise measurement of positions and motion of celestial bodies. The path the star makes in the sky is seen to wobble due to the gravitational effects of its exoplanet.

Radial velocity and astrometric detection rely on dynamics. Transit photometry relies solely on the light measures from a star. When the conditions are just right, an exoplanet will pass in front of its star relative to the observer. This event is observed as a dip in the amount of light measured over time from a star. The Kepler mission used this method by staring at a patch of sky for long periods, measuring the light from stars. On rare occasions, astronomers have used transits to observe exoplanetary atmospheres as the starlight passes through them. A different detection method, gravitational microlensing, uses the light of another star. According to general relativity, the gravitational field of a massive body warps the spacetime surrounding it, which can have the effect of an optical lens. When a background star passes behind a foreground star relative to an observer, the background star is magnified. An exoplanet orbiting the foreground star measurable augments this effect.

Exoplanets can also be directly imaged. This has been attempted at multiple wavelengths, such as in the infrared where exoplanets are expected to have stronger emissions. The difficulty of direct imaging stems from the fact that exoplanets can billions of times dimmer than their stars [3]. Coronagraphy addresses this by introducing a physical apparatus in a telescope to block the light of the star. This allows a detector to image an exoplanet via the starlight reflected off of it. The benefit of this method is the greater range of systems it is sensitive to. The previous methods have strong dependencies on parameters such the mass of the exoplanet or the orientation of its orbit relative to the observer. Coronagraphy is able to directly imaging a wider range of sources and is continually improving in capability.

*Depending on which branch of government you ask, the missions may or may not be terminated.

†More precisely, the star and planet orbit their mutual center of mass.

2 WFIRST Coronagraphy

The Wide-Field Infrared Survey Telescope will detect and characterize nearby exoplanets down to the mass and size of Jupiter using the CGI, which is currently under development. The design of the instrument is driven by the performance metric of how many such sources it can observe in its lifetime. This science yield ultimately depends on the integration time needed to obtain a statistically significant detection defined by the desired signal-to-noise ratio (SNR). The instrument will have two modes: direct imaging and spectroscopy. The first mode is straightforward; the incoming light, after passing through all the telescope and coronagraph elements, is recorded by a detector. The second mode utilizes an integral field spectrometer (IFS), which essentially breaks up an image into distinct, spatially resolved spectra. For both modes, the same charge-coupled device (CCD) detector is used [1].

Broadly speaking, there are three sources of photons to be modelled in a coronagraph measurements. First are photons from the target, which is the light reflected off the exoplanet from its host star. Next is the light from speckle effect, which are the residual traces of the blocked out star, as seen in Figure (1). Last are background noise photons from various sources. When any photons arrive at the CGI detector, they produce electrons which are counted pixel by pixel to produce an image. The electron count rates from the three photon sources are termed C_p, C_{sp}, and C_b, respectively. These depend on the actual photon rates and the parameters of the observatory, which themselves may depend on operating wavelength λ and mission time t'.

A successful observation will have an integration time long enough to ensure that the strength of the signal from the exoplanet point source divided by the strength background noise is $\geq$ SNR. Physically, the exoplanet signal strength is a function of the flux from the exoplanet, F_p, given by:

$$F_p = F_* \, F_{p/*} = F_0 \, 10^{-\frac{2}{5}(m_* + \Delta m)}. \qquad [\mathrm{ph\,s^{-1}\,m^{-2}}] \quad (1)$$

Here m_* is the apparent magnitude of the host star, and Δm is the difference in magnitude between the host star and exoplanet. This is product of the flux from the star, F_* and the difference in flux between the two sources, $F_{p/*}$:

$$F_* = F_0 \, 10^{-\frac{2}{5} m_*}, \qquad [\mathrm{ph\,s^{-1}\,m^{-2}}] \quad (2)$$

and

$$F_{p/*} = p \left(\frac{R_p}{a} \right)^2 = 10^{-\frac{2}{5} \Delta m} \qquad [\mathrm{ph\,s^{-1}\,m^{-2}}] \quad (3)$$

respectively. F_0 is the zero magnitude flux from the star. There are two ways of calculating $F_{p/*}$: using the physical parameters of the exoplanet system (albedo p, planet radius R_p, and semi-major axis a); or using the difference in magnitude Δm. The rate of photons from an exoplanet is:

$$r_p(\lambda) = F_p \, f_{SR} \, A_{col} \, \eta'_{QE}(\lambda) \, \tau_{PS}(\lambda). \qquad [\mathrm{ph\,s^{-1}}] \quad (4)$$

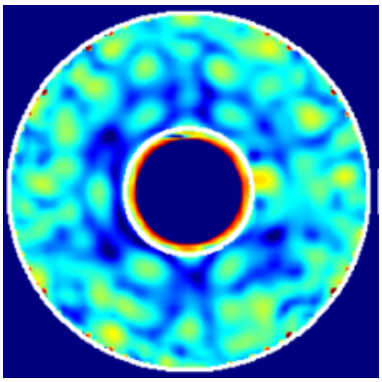


Figure 1: Simulated CGI measurement displaying residual speckle effect from star. [4]

f_{SR} is the fraction of light in the signal region, which is unity for the direct imaging mode and $(R\,BW)^{-1}$ for IFS mode, where R is spectral resolution and $BW = \Delta\lambda/\lambda$ is the fractional bandwidth. A_{col} is the collecting area of the detector in m^2. τ_{PS} is the throughput of light from the point source, a value between 0 and 1 representing the percent of incoming light arrives at the detector due to losses from reflection, transmission, and diffraction with surfaces:

$$\tau_{PS}(\lambda) = \tau_{core}(\lambda)\,\tau_{rf,tr}(\lambda), \tag{5}$$

where τ_{core}, representing diffraction effects, is the ratio of light on the detector's point spread function (PSF) core to incident light on the primary mirror of the telescope, and $\tau_{rf,tr}$ represents losses from reflections and transmissions due to mirror coatings and filters. The last term in Equation (4), η_{QE}, is the original quantum efficiency of the CCD, which represents the detector's sensitivity to light at a given wavelength. As the mission will progress the space environment will negatively impact the detector, and quantum efficiency becomes:

$$\eta_{QE}(\lambda, t') = \eta'_{QE}(\lambda)\,\eta_{PC}\,\eta_{HP}(t')\,\eta_{CR}(t')\,\eta_{NCT}(t'), \tag{6}$$

where η_{PC} is the photon counting efficiency, η_{HP} represents hot pixel degradation, η_{CR} represents damage due to cosmic rays, and η_{NCT} is the net charge transfer efficiency. The above equations produce the electron count rate from the exoplanet:

$$C_p(\lambda, t') = f_{SR}\,F_p\,A_{col}\,\tau_{PS}(\lambda)\,\eta_{QE}(\lambda, t'). \qquad [e^-\,\mathrm{s}^{-1}] \tag{7}$$

The electron count rate from speckling effects has additional dependencies on the characteristics of the observatory:

$$C_{sp}(\lambda, t') = f_{SR}\,F_*\,A_{col}\,\tau_{rf,tr}(\lambda)\,\eta_{QE}(\lambda, t')\,\frac{C_{CG}(\lambda)}{k_{pp}}\,I_{pk}(\lambda)\,m_{pix,CG}(\lambda). \qquad [e^-\,\mathrm{s}^{-1}] \tag{8}$$

C_{CG}/k_{pp} is the raw contrast achieved by the coronagraph (C_{CG}) divided by a post-processing factor (k_{pp}), resulting in an greater effective contrast. I_{pk} is the fraction of incident photons that reach the peak of the detector's PSF [2]. Lastly, $m_{pix,CG}$ is the number of pixels sampling the core of the PSF:

$$m_{pix,CG}(\lambda) = A_{PSF}(\lambda)\left(\frac{D_{PM}}{\lambda_d\,k_s}\right)^2\left(\frac{\pi}{180\,\times\,3600}\right)^2, \tag{9}$$

where A_{PSF} is the area of the PSF in square measured in square arcseconds, D_{PM} is the diameter of the primary mirror measured in m, λ_d is the design wavelength of the coronagraph and k_s is its intrinsic sampling.

There is a similar value for the number of pixels within the region of interest where the target is located on the detector:

$$m_{pix,pl}(\lambda) = \begin{cases} A_{PSF}(\lambda)\left(\frac{\lambda}{\lambda_d}\right)^2\left(\frac{2\,D_{PM}}{\lambda_c}\right)^2, & \text{if Imaging;} \\ L_{PSF}\left(\frac{\lambda}{\lambda_c}\right)^2 n_{col}\,n_{row}, & \text{if Spectroscopy.} \end{cases} \tag{10}$$

Here, λ_c is the wavelength corresponding to Nyquist sampling (5.08×10^{-7} m for imaging, 6.60×10^{-7} m for spectroscopy), $L_{PSF} = 5$ is the number of spectroscopy lenslets sampled by the PSF core, and $n_{col} = 2$ and $n_{row} = 2$ are the numbers of pixels in the spectral and spatial dimensions for each lenslet.

To calculate the electron count rate from background noise sources, there are additional photon rates to take into account. First, the speckle photon rate is very similar to Equation (8), the speckle electron count rate:

$$r_{sp}(\lambda) = f_{SR}\,F_*\,A_{col}\,\tau_{rf,tr}\,\eta'_{QE}(\lambda)\,C_{CG}(\lambda)\,I_{pk}(\lambda)\,m_{pix,CG}(\lambda). \qquad [\mathrm{ph\,s}^{-1}] \tag{11}$$

Another source of noise is from zodiacal light, which is reflected from dust within the solar and extra-solar systems (referred to as local zodi and exo zodi, respectively). The fluxes from these sources are F_{lzo} and F_{ezo} (in $\mathrm{ph\,s^{-1}\,m^{-2}}$), and the photon rates are:

$$r_{ezo}(\lambda) = f_{SR}\,F_{ezo}\,\tau_{rf,tr}(\lambda)\,\tau_{occ}(\lambda)\,\eta'_{QE}(\lambda)\,A_{col}\,A_{PSF}(\lambda) \qquad [\mathrm{ph\,s}^{-1}] \tag{12}$$

and

$$r_{lzo}(\lambda) = f_{SR}\,F_{lzo}\,\tau_{rf,tr}(\lambda)\,\tau_{occ}(\lambda)\,\eta'_{QE}(\lambda)\,A_{col}\,A_{PSF}(\lambda), \qquad [\mathrm{ph\,s}^{-1}] \tag{13}$$

where τ_{occ} is the fraction of light passing through the first pupil in the coronagraph that reaches the detector.

There are four additional charge sources of noise related to CCD operational characteristics. Dark noise is the phenomenon whereby thermal effects produce charges unrelated to incoming light. The dark noise rate is:

$$C_{DN}(\lambda, t') = \mathrm{ENF}^2\, i_d(t')\, m_{pix,pl}(\lambda), \qquad [e^-\,\mathrm{s}^{-1}] \quad (14)$$

where ENF is excess noise factor of the detector (usually equal to unity) and i_d is the dark current (in $e^-\,\mathrm{s}^{-1}$). Clock induced charges are spurious electrons generated during clocking (when charges are counted). The associated rate is:

$$C_{CIC}(\lambda) = \mathrm{ENF}^2\, k_{CIC}\, \frac{m_{pix,pl}(\lambda)}{t_f}, \qquad [e^-\,\mathrm{s}^{-1}] \quad (15)$$

where k_{CIC} is a constant and t_f is the exposure time per frame (measured in s). Next, there are charges created by stray light, termed as luminescent background, with a rate of:

$$C_{lum}(\lambda, t') = \mathrm{ENF}^2\, \eta_{QE}(\lambda, t') \left[625\, m_{pix,pl}(\lambda) \left(\frac{s_{pix}}{0.2\,\mathrm{m}}\right)^2 + 1.25\,\pi\, \frac{m_{pix,pl}(\lambda)}{n_{pix}}\right], \quad [e^-\,\mathrm{s}^{-1}] \quad (16)$$

where $s_pix = 1.30 \times 10^{-5}\,\mathrm{m}$ is the pixel size and $n_pix = 1024^2$ is the number of pixels in the detector. Lastly, the read noise rate, representing charges produced during charge amplification and readout, is given by:

$$C_{RN}(\lambda) = \frac{m_{pix,pl}(\lambda)}{t_f} \left(\frac{k_{RN}}{k_{EM}}\right)^2, \qquad [e^-\,\mathrm{s}^{-1}] \quad (17)$$

where k_{RN} and k_{EM} are constants. Additional constants include k_{sp} and k_{det}. With these values, the total noise electron count rate is:

$$C_b(\lambda, t') = \mathrm{ENF}^2 \left[\begin{array}{l} r_p(\lambda) + k_{sp}\,[r_{sp}(\lambda) + r_{ezo}(\lambda)] + k_{det}\,[r_{lzo}(\lambda) + \\ C_{DN}(\lambda, t') + C_{CIC}(\lambda) + C_{lum}(\lambda, t') + C_{RN}(\lambda)] \end{array}\right]. \qquad [e^-\,\mathrm{s}^{-1}] \quad (18)$$

Finally, the integration time necessary to reach a desired SNR is given by:

$$t_{\mathrm{SNR}}(\lambda, t') = \frac{\mathrm{SNR}^2\, C_b}{C_p^2(\lambda, t') - \mathrm{SNR}^2\, C_{sp}^2(\lambda, t')}. \qquad [\mathrm{s}] \quad (19)$$

3 Modelling Procedure

EXOSIMS is an object-oriented software framework implemented in Python for exoplanet observation simulation. The purpose is to allow users to generate large ensembles of simulations to optimize an observation plan. An EXOSIMS mission is made up of stand-alone modules [5]:

- `StarCatalog` : Includes detailed information about potential target stars drawn from general databases such as SIMBAD, mission catalogs such as Hipparcos, or from existing curated lists specifically designed for exoplanet imaging missions.
- `PlanetPopulation` : Encodes the probability density functions of all required planetary parameters, both physical and orbital, as well as generating functions to return samples of these parameters.
- `PlanetPhysicalModel` : Contains models of the light emitted or reflected by planets in the wavelength bands under investigation by the current mission simulation.
- `OpticalSystem` : Contains all of the necessary information to describe the planet signal and background noise fluxes at the image plane of all relevant instruments, and calculate the required integration time for a given observation.
- `ZodiacalLight` : Contains methods to calculate the zodiacal light surface brightness of local Zodi, extrasolar Zodi, star specific minimum Zodi, and star specific maximum Zodi.

- `BackgroundSources` : Provides density of background sources for a given target based on its coordinates and the integration depth.
- `PostProcessing` : Encodes the effects of post-processing on the data gathered in a simulated observation, and the effects on the final contrast of the simulation.
- `Completeness` : Takes in information from the Planet Population module to determine initial completeness and update completeness values for target list stars when called upon.
- `TargetList` : Takes in information from the `StarCatalog` , `OpticalSystem` , `ZodiacalLight` , `PostProcessing` , `BackgroundSources` , `Completeness` , `PlanetPopulation` , and `PlanetPhysicalModel` modules to generate the target list for the simulated survey.
- `SimulatedUniverse` : Instantiates the `TargetList` module and creates a synthetic universe by populating planetary systems about some or all of the stars in the target list.
- `Observatory` : Contains all of the information specific to the space-based observatory not included in the `OpticalSystem` module.
- `TimeKeeping` : Is responsible for keeping track of the current mission time, exoplanet observation time, and observing blocks.

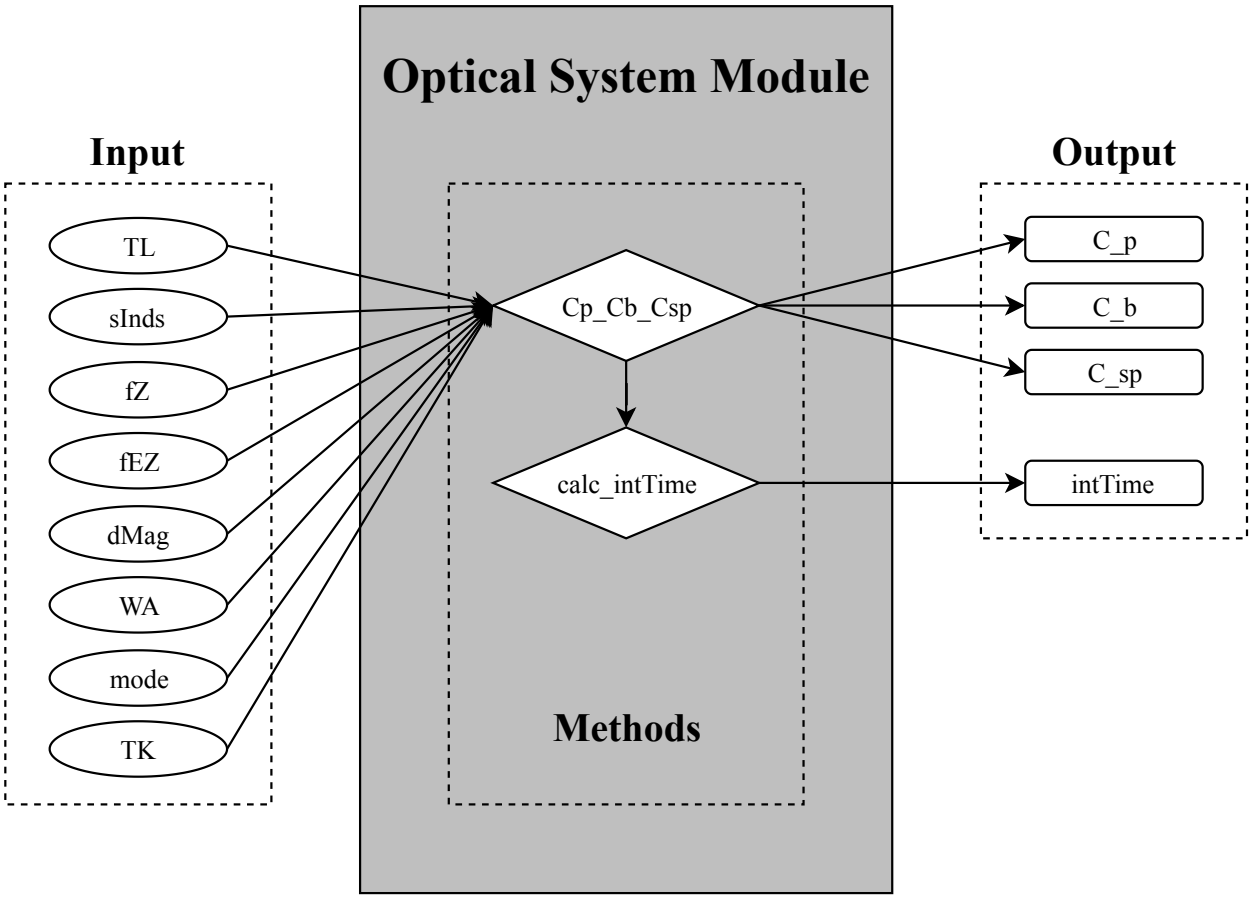


Figure 2: Depiction of select `OpticalSystem` module methods including input and output. [5]

- `SurveySimulation` : Performs a specific simulation based on all of the input parameters and models.
- `SurveyEnsemble` : Runs multiple simulations.

The above modules exist both as prototype class objects and more as detailed instances. A user is able to specify, among many observatory parameters, which model of these classes to use. The focus of this project is to implement a more complex and accurate `OpticalSystem` module. The design is based on the research of Dr. Bijan Nemati, a research scientist working on the CGI development. An earlier, less exact module named `Nemati` exists within EXOSIMS. Through this project, a new module named `Nemati_2019` based on current understandings of the CGI is developed, which inherits a `Nemati` object, which in turn inherits the `OpticalSystem` prototype. The newer version will provide more accurate integration times.

The method `calc_intTime` which calculates integration times takes as inputs:

- `TL` (`TargetList` module): `TargetList` class object.
- `sInds` (integer array): Integer indices of the stars of interest.
- `fZ` (`astropy` [‡] quantity array): Surface brightness of local zodiacal light in units of arcsecond^{-2}.
- `fEZ` (`astropy` quantity array): Surface brightness of exo-zodiacal light in units of arcsecond^{-2}.
- `dMag` (float ndarray): Differences in magnitude between planets and their host star.
- `WA` (`astropy` quantity array): Working angles of the planets of interest in units of arcsec.
- `mode` (dict): Selected observing mode.
- `TK` (`TimeKeeping` module): `TimeKeeping` class object.

and its outputs are:

- `C_p` (`astropy` quantity array): Planet signal electron count rate in units of s^{-1}.
- `C_b` (`astropy` quantity array): Background noise electron count rate in units of s^{-1}.
- `C_sp` (`astropy` quantity array): Residual speckle spatial structure (systematic error) in units of s^{-1}.

The `calc_intTime` method from `Nemati` was not overridden, since it was an accurate implementation of Equation (19). Rather, the method `Cp_Cb_Csp` was re-implemented in `Nemati_2019` . This method takes the same inputs as `calc_intTime` , and is called by that method. This relation is shown in Figure (2). When a class method is not implemented in a module which is used in a mission simulation, the method is accessed from the object the module inherits, such as a prototype.

To carry this out, the equations describing how the CGI works (a subset of which are the presented above) was adapted as Python code in the EXOSIMS framework. The first step was to completely replicate Dr. Nemati's work as code. This involved three kinds of variables: the outputs (`C_p` , `C_b` , and `C_sp`), intermediate calculations, and input parameters. Inputs are either constants or functions of wavelength `lam` $\equiv \lambda$ and/or working angle `WA` (a coronagraph has an inner and an outer working angle, which define the angular field of view of the instrument). The result of this process was a functional model of the CGI.

The next stage involved integrating this model within the EXOSIMS framework. Many intermediate calculations were replaced by input parameters accessible within a mission simulation. Remaining inputs

[‡] `astropy` is a free, community developed Python library with useful utilities for astronomy.

were matched to existing parameters within EXOSIMS. These parameters are specified in a `.json` scriptfile by which a mission simulation is created. Constant inputs are given as floats and functions are given by paths to `.fits` files, which are interpolated in the `OpticalSystem` prototype.

The final step is to address all of the effects that constructing `Nemati_2019` had on EXOSIMS as a whole. The most pervasive change was adding `TK` as an argument to `Cp_Cb_Csp` . Because this method is called throughout the software, `TK` had to be tracked up through numerous classes and included where appropriate. Additionally, the `OpticalSystem` methods `calc_dMag_per_intTime` and `ddMagdt` need to be overridden. The former method takes an array of integration times (`intTimes`) along with other inputs, calls `Cp_Cb_Csp` , and calculates the original `dMag` $\equiv \Delta m$. The second method finds the derivative of `dMag` with respect to `intTimes` ($d\Delta m/dt_{SNR}$). The difficulty faced in these steps is the non-analytic nature of the relationship between Δm and t_{SNR}:

$$t_{SNR} = \frac{\mathrm{SNR}^2\,\mathrm{ENF}^2\left[F_*\,10^{-\frac{2}{5}\Delta m}\,f_{SR}\,A_{col}\,\tau_{PS}\,\eta_{QE} + k_{sp}\left(r_{sp}+r_{ezo}\right) + k_{det}\left(r_{lzo}+C_{DN}+C_{CIC}+C_{RN}+\mathrm{ENF}^2\,\eta_{ph}\,\eta'_{QE}\,\eta_{HP}\,\eta_{CR}\,\mathrm{CTE}\right.\right.}{\left.\left.max\left\{0,\ min\left\{(1-0.142\,t_{MF}),\left(1+t_{MF}\left[-0.142+3.42\left(\frac{t_f}{m_{pix,pl}}\left[F_s\,10^{-\frac{2}{5}\Delta m}\,f_{SR}\,A_{col}\,\tau_{PS}\,\eta'_{QE}+r_{sp}+r_{ezo}+r_{lzo}\right]-0.089\right)\right]\right)\right\}\right\}\right]}$$
$$F_*^2\,10^{-\frac{4}{5}\Delta m}\,(f_{SR}\,A_{col}\,\tau_{PS}\,\eta'_{QE}\,\eta_{PC}\,\eta_{HP}\,\eta_{CR})^2\left(\mathrm{CTE}\ max\left\{0,\ min\left\{(1-0.142\,t_{MF}),\right.\right.\right.$$
$$\left.\left.\left.\left(1+t_{MF}\left[-0.142+3.42\left(\frac{t_f}{m_{pix,pl}}\left[F_*\,10^{-\frac{2}{5}\Delta m}\,f_{SR}\,A_{col}\,\tau_{PS}\,\eta'_{QE}+r_{sp}+r_{ezo}+r_{lzo}\right]-0.089\right)\right]\right)\right\}\right\}\right)^2+\mathrm{SNR}^2\,C_{sp}^2} \tag{20}$$

where new variables represent additional mission simulation characteristics.

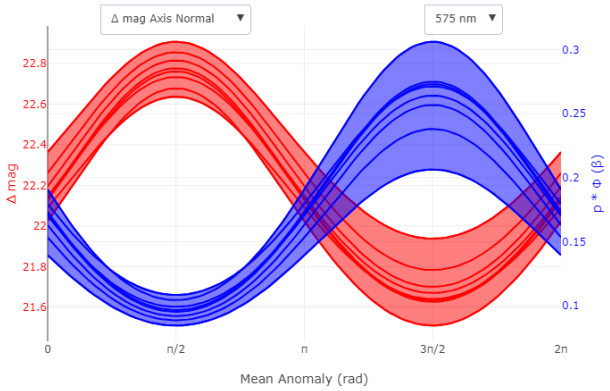


Figure 3: Δm and $p\,\phi(\beta)$ vs. mean anomaly ($\phi(\beta)$ is the phase function). This shows how the observability of an exoplanet changes as it orbits its host star. The widths of the two functions incorporate models produced assuming different inclinations of the exoplanet system. [6]

4 Results

The method `Cp_Cp_Csp` implemented in `Nemati_2019` successfully represents the current best understood model of the WFIRST CGI. The new versions of `calc_dMag_per_intTime` and `ddMagdt` are not yet complete. With successful validation testing complete, in the next step `Nemati_2019` module and associated changes to EXOSIMS will merge with the main branch of the software.

A secondary result of this project was employing `calc_intTime` using the new `Cp_Cb_Csp` method to calculate integration times for actual exoplanets. A script was written to read in the modeled orbit data of exoplanets from the SIOS Imaging Mission Database [6]. For this, Δm, m_s, working angle, star name, and star spectral type were used to fill the necessary mission simulation parameters. Integration times achieving an SNR ≥ 10 were calculated and recorded. Figure (3) shows Δm and phase function throughout the orbit of an exoplanet; the integration times will be included in the future as a third function.

5 Conclusion

The Wide-Field Infrared Survey Telescope will feature a technology demonstration coronagraph to perform high contrast direct imaging of exoplanets. Because these sources are so difficult to detect and the observatory and instrument are so complex, it is important to fully understand how this science will be performed during the mission. This is the purpose of the Exoplanet Open-Source Imaging Mission Simulator: it allows scientists and engineers to accurately model exoplanet measurements to estimate science yields and schedule observations. Of vital importance is integration time, which is the necessary time spent observing a target to yield a desired signal-to-noise ratio.

The goal of this project was to produce an accurate representation of integration time for the WFIRST coronagraphic instrument in EXOSIMS. This task was achieved, what remains is to merge the updates and changes with the main software. The code produced was successfully used to calculate integration time models for a number of known exoplanet systems. This functionality will allow EXOSIMS users to explore instrument and observatory designs and will hopefully benefit ongoing mission development.

References

[1] Traub, W. A. et. al., *Science Yield Estimate with the Wide-Field Infrared Survey Telescope Coronagraph*, J. Astron. Telesc. Instrum. Syst. 2(1) 011020 (18 March 2016).

[2] Nemati, B. et. al., *Sensitivity of the WFIRST Coronagraph Performance to Key Instrument Parameters*, Proc. SPIE 10400, Techniques and Instrumentation for Detection of Exoplanets VIII, 1040007 (1 September 2017).

[3] Perryman, M., *The History of Exoplanet Detection*, Astrobiology 2012 12:10, 928-939 (18 October 2012).

[4] Nemati, B., *WFIRST Coronagraph Simulations and Performance Estimates*, Jet Propulsion Laboratory, California Institute of Technology, Stanford Meeting 10, (7 December 2017)

[5] Keithly, D. et. al., *Exoplanet Open-Source Imaging Mission Simulator (EXOSIMS) Interface Control Document*, Sibley School of Mechanical and Aerospace Engineering (Last Update: 25 March 2019).

[6] SIOS Imaging Mission Database, *Planet Detail for RR Cae b*, SIOSlab. `https://plandb.sioslab.com/plandetail.php?name=RR+Cae+b`