Automating Noise Reduction Procedures for Exoplanet Research

Alex Spencer

A senior thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Bachelor of Science

Denise Stephens, Advisor

Department of Physics and Astronomy

Brigham Young University

ABSTRACT

Automating Noise Reduction Procedures for Exoplanet Research

Alex Spencer
Department of Physics and Astronomy, BYU
Bachelor of Science

Information gained from analyzing exoplanets should answer questions about planetary and solar system formation. At Brigham Young University, the existence of exoplanets are confirmed using the transit detection method. However, information about these objects of interest are shrouded by the telescope's inherent noise. In order to analyze the desired information, noise reduction procedures must be applied manually through command-line methods. This reduction process is slow and tedious, causing the sought-after data to often go untouched for long periods of time. To circumvent this, a program—called the pipeline—was further developed to perform these procedures automatically. The pipeline calls C-style script files to complete each action automatically, analyzing and responding to the discrepancies in each data set. The pipeline performs noise reduction quickly, consistently, and reliably. When compared to student researchers performing reduction tasks manually, the pipeline displays improvements in the processed data. Most importantly, the pipeline allows student researchers more time to focus on analyzing results or to work on other projects.

Keywords: Exoplanet, transit, IRAF, noise reduction, automation, Barycentric Julian Date, calibration frames, KELT

ACKNOWLEDGMENTS

I’d like to acknowledge and thank Dr. Denise Stephens for her guidance, countless solutions, patience, and motivation.

I’d also like to thank my friend and colleague Ian Clark for listening to every roadblock I encountered, my parents for their unfailing encouragement, and my countless other supporters.

I lastly acknowledge and thank the department of Physics and Astronomy at Brigham Young University for the opportunities, ability to research, and for my experiences as an undergraduate.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Observational data from telescopes can be used for a wide variety of research topics. However, the desired information from these raw images is obscured by noise—or unwanted variations in the recorded data—inherent in the telescope system. Before analysis can be performed, the noise in these images must be reduced. Noise reduction procedures are performed in Image Reduction and Analysis Facility (IRAF). The command-line operations of this program gives the user a remarkable level of control, but it is also slow and tedious to operate. As a result, noise reduction becomes a large obstruction in the research capacity of any astronomy project. To resolve this, I developed a program that could automatically perform noise reduction procedures.

1.2 Transiting exoplanets

The existence of exoplanets has been theorized for centuries (Croswell 1997), but only recent advancements in technology have allowed exoplanets to be confirmed and studied. An exoplanet is a planet orbiting a star that is not our sun. In the sixteenth century, an Italian philosopher and supporter

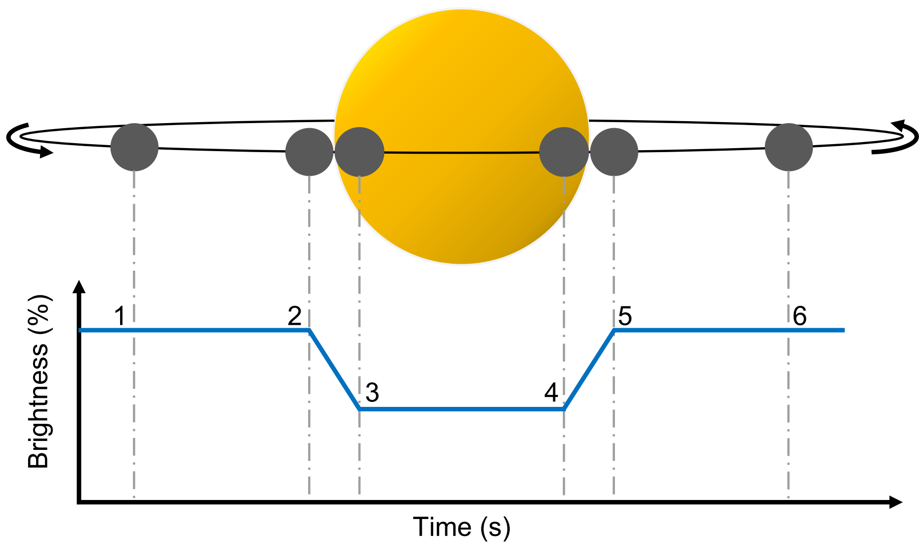


Figure 1.1 Schematic of a light curve found while observing a transit. Light curves can be used to gain information about the exoplanet. Points 2 to 3 correlates closely with the diameter of the exoplanet. If the exoplanet is larger—relative to its star—it will block more starlight and cause a greater dip in brightness to be measured. These facts allow the exoplanet to be studied without observing the planet itself.

of the Copernican model—Giordano Bruno—wrote theories about a multiplicity of worlds like our own (Croswell 1997). In the mid twentieth century, searching for exoplanets became a common research topic. However, until the advent of the charged-coupled device (CCD), no exoplanets were officially confirmed. The use of CCD chips in telescopes allowed for greater advancements into Astronomy research, and—in the mid 1990s—the first exoplanets were discovered (Haswell 2010). Since then, due to further technological advancements and missions like the Kepler space telescope, many exoplanets have been confirmed using a variety of detection methods (Carroll & Ostlie 2017).

One of the most popular exoplanet detection methods is using transit detection. In a transit, an exoplanet passes in between its star and the Earth. The exoplanet partially eclipses its star, causing a change in the light observed, as shown in Fig. 1.1. The small, gray planet passes in front of its star between points two and five. As a result, the brightness of the star that is being observed drops slightly. This change in detected light is called a light curve, and it contains various information

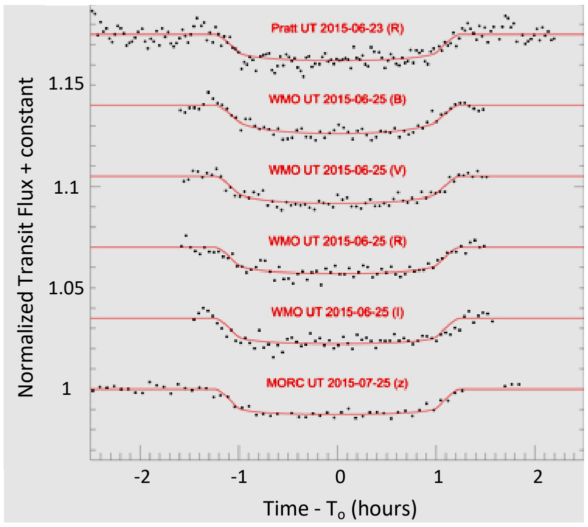


Figure 1.2 Light curve data of exoplanet KELT-16b, modified from Oberst et al. (2017). This shows data from BYU's Pratt Observatory, BYU's West Mountain Observatory (WMO), and University of Louisville Moore Observatory (MORC). The black dots are the measured light output while the red line indicates a best fit transiting exoplanet. The transit results in a light curve which—when combined with other observations—is used to make predictions about the exoplanet.

about the exoplanet. For example, Fig. 1.1 suggests that the points between two and three contain the exoplanet's diameter.

Though its less concise, observational data can be used to measure an exoplanets diameter. In Fig. 1.2, an example is given of published transiting exoplanet data. The light curves shown are modified from Oberst et al. (2017) for the exoplanet KELT-16b. The y-axis has been normalized, and then shifted vertically so that curves from a variety of nights can be displayed together. The top curve was taken here at Brigham Young University (BYU) in the Orson Pratt Observatory (OPO). These observations display noise—variations from the predicted location of the plotted data points—even after extensive noise reduction procedures have been performed. The remaining noise is likely caused by atmospheric disruptions or error in the observational systems. Despite this noise, it is possible to find a consistent light curve across multiple observations from multiple locations. Each light curve shows a consistent drop in the star's light, just as in Fig 1.1. Despite the additional noise, these light curves can be used to find the exoplanet's diameter.

A leading objective in transit detection is to learn about the exoplanet's composition, which is best predicted by knowing the exoplanet's density. Through light curve analysis, the diameter—and the radius—of the planet is found, which can be used to obtain the volume. Since planets are roughly spherical, the volume is equal to $\frac{4}{3}\pi r^3$, where r is the predicted radius. Through the addition of another exoplanet detection method—radial velocity—the mass of the exoplanet can be obtained.

Radial velocity is an exoplanet detection method that requires spectroscopy. In a two-body orbiting system, both bodies orbit the center of mass point. As a result, when a star has an orbiting planet, the star will also be moving in a small orbit. This means that at some points in its orbit, the star is moving towards Earth, while at other times it is moving away. The star's motion results in a Doppler shift of the output light. Using spectroscopy, it is possible to measure these shifts and relate them to the orbital velocity of the star. The momentum—mass times velocity—should be equal for both bodies in orbit. The star's classification on the H-R diagram gives a predicted mass while the orbital period of the exoplanet approximates the exoplanet's orbital velocity. Thus, the exoplanet's predicted mass is equal to its star's momentum divided by the exoplanet's orbital velocity (Carroll & Ostlie 2017). By dividing the volume into this predicted mass, an average density for the exoplanet is obtained. This density identifies the main composition of the planet being observed. In order of increasing density, the composition may primarily be gaseous, icy, or rocky (Haswell 2010).

Transiting exoplanets are favorable to study since they are easy to examine, simple to analyze, and are well-proven. Neither transit detection or radial velocity measurements require observing the planet directly. As a result, observing transiting exoplanets is useful for smaller observatories. Analysis of a transit is exceptionally simple, due to the increasing number of software programs available. AstroImageJ (AIJ), for example, can extrapolate light curves and predict details about the transiting exoplanet from limited observations (Kielkopf Accessed Apr 9, 2019). Notwithstanding this simplicity, transit analysis is well-proven—largely due to the Kepler space telescope utilizing it.

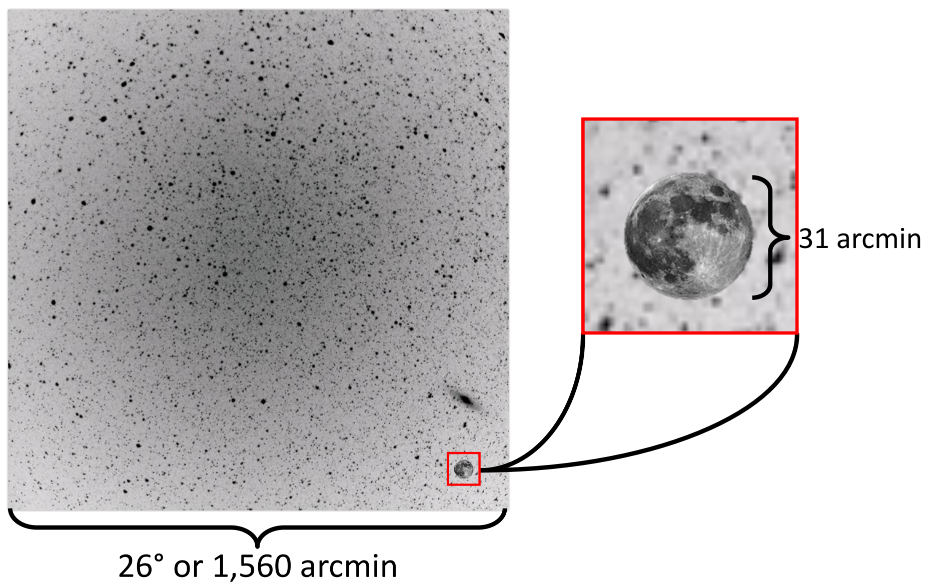


Figure 1.3 The field of view (FOV) of KELT is 26° by 26°. The moon is shown here to scale. For comparison, our telescope in OPO at BYU has a 16.6 by 16.6 arcminutes FOV, which is nearly half the size of the full moon pictured here.

As a result, this method has confirmed more exoplanets than any other exoplanet detection method (Haswell 2010). Because transit analysis is well-proven and easy to perform, many exoplanet surveys utilize these methods.

1.3 The KELT team

One project using transit detection method is the Kilodegree Extremely Little Telescope (KELT), which detects potential exoplanet candidates using a wide-angle field of view, long observation times, and a fast Fourier transform (FFT). Despite it being about the size of a soda can, the KELT telescope is very powerful. It is able to image a large portion of the night sky with a significant 26° by 26° field of view (FOV). In arcminutes (denoted by an apostrophe), this FOV is 1560' by 1560'. For comparison, the moon's angular size is 31'. The difference is shown in Fig. 1.3, with the moon being placed to scale on a frame from KELT (Pepper et al. Accessed Jan 15, 2019).

The wide-angle field of KELT is used to observe a large portion of the sky at once. Each star on its frame is given an identifier number. The brightness of each identified star is pulled out from every image and is saved in tabular form on KELT's servers. This tracks the time-varying brightness of the stars in that field. The brightness measurements are then examined using a FFT—which transforms the brightness data into frequency space. The FFT checks for any periodic changes in light, indicative of an exoplanet. For further information of this process, see Pepper et al. (2007). When KELT finds evidence of recurring fluctuations in a star's brightness, information about an exoplanet candidate is sent to the KELT follow-up network.

Despite the many advantages of the KELT telescope, a follow-up network is necessary in order to confirm exoplanets. The FOV used by KELT allows them to image many stars at once, improving the chances of observing a light curve caused by a planet. However, this wide-angle FOV is difficult to focus on their CCD chip. As a result, light from stars can blend together on the same pixel, making it impossible to tell which star has caused the light curve. This effect can be observed in the expanded image of the moon in Fig. 1.3. The stars—imaged in black—become blurred and unclear. The blending of light may often result in a false reading, called a false positive. False positives are generally caused by dim variable or binary star's light blending with a brighter star. The brightness measured by KELT appears to dip less for this more luminous star, resulting in an exoplanet-like light curve. For further information regarding false positives, see Collins et al. (2018). Due to the possibilities of star light blending caused by their wide-angle field, KELT requires a team of researchers to follow up on potential exoplanet candidates.

1.4 Previous work at BYU

Researchers at Brigham Young University (BYU) follow up on potential exoplanet candidates from KELT by observing the object of interest during theorized times of transit. As members of KELT's follow-up network, we receive information about possible exoplanets from KELT's observations. If this object is visible from our observatory, we'll attempt to image it during the projected times of transit. With the use of a CCD chip camera, we can detect changes in light of a target star as small as 1.0% of the star's initial brightness (Haswell 2010). After performing noise reduction procedures, the images are analyzed using AIJ. Our findings—whether the candidate is an exoplanet or a false positive, the correct period of the candidate, and our observed light curve—are sent back to the KELT team. Though our list of tasks in the KELT team is relatively small, we must sift through an ever increasing volume of exoplanet candidates and observational data.

Being members of the KELT team requires sorting through a large amount of data from the KELT telescope—as well as from our own observations. The KELT telescope has found many potential exoplanet candidates, making it difficult to work on all of them. Every object that we do observe results in hundreds of images along with dozens of calibration images. These images need to be processed before they can be analyzed. So, before using AIJ, each image must be prepared using noise reduction procedures. The noise reduction process is slow and tedious, and—as a result—many nights of data sit untouched for long periods of time.

1.5 Automatic noise reduction

In order to eliminate this concern, I have created a program that performs noise reduction processing automatically. This program is called the pipeline—a programming term used to describe a system in which input data is transferred logically through a list of linear operations, each of which leads into the next (Patterson 2017). Thus, our raw images are input into the program and "pipelined" through each of the procedures of noise reduction.

The pipeline has proven itself profoundly beneficial in our exoplanet research. The pipeline performs noise reduction faster than most student researchers. The resulting data is processed in an identical manner, allowing the analysis to be more uniform and more reliable. Because the raw data is processed quicker, the completed analysis can be returned to the KELT team sooner. Moreover, the pipeline was created using only free software—IRAF and Python—allowing it to be convenient and publicly accessible. The best result, however, is that the pipeline allows student researchers to focus more time to analyze data and to work on other research projects. The pipeline has significantly improved our research capabilities.

1.6 Overview

This thesis is organized as follows. First, chapter 2 discusses the methods leading to this automated program. The required procedures in our noise reduction system are explained. The programming capabilities of IRAF as they pertain to the creation of the pipeline are then described. Next, the resolution to some issues encountered are outlined. Following this, chapter 3 reports the finished program's results. The pipeline's timing and resulting data sets are compared against student researchers. The resulting program processes images quicker, consistently, and more reliably.

Chapter 2

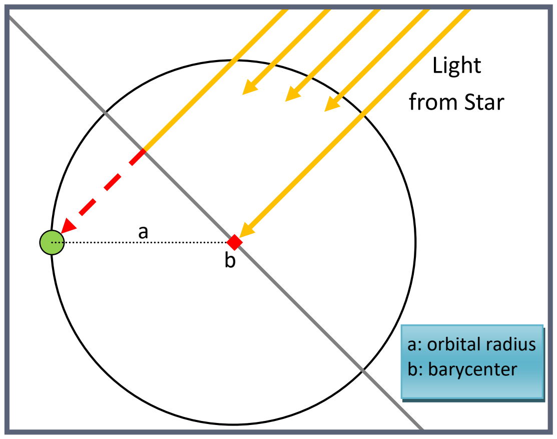
Methods

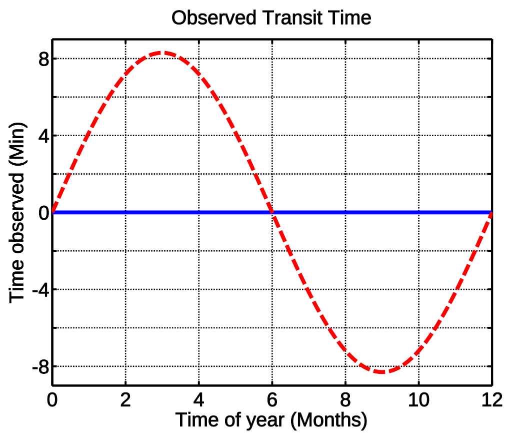
2.1 Required procedures

As members of the KELT team, we are required to complete a list of noise reduction procedures on each data set before analysis. By performing these well-proven steps, the data returned by team members is standardized and can be studied collectively. The most important steps include changing the time stamp on each image to Barycentric Julian Date (BJD) and applying calibration frames—zero, dark, and flat frames—to the object frames.

2.1.1 Converting to BJD

First, we must convert the time of exposure to Barycentric Julian Date (BJD). On Earth, events are measured using local times, which can easily be converted into Julian Date (JD). For a large network like KELT, this creates a discrepancy in our data. The team must use a universal time standard. It can't, however, be an Earth-based time standard, because the Earth moves. Light from the same source must travel different distances to reach Earth at unique points in its orbit. An example of this is shown in Fig. 2.1(a). Since stars are very distant, their light is approximately plane parallel, as





(a) **(b)**

Figure 2.1 (a) Schematic comparing light's arrival at the barycenter versus the Earth. As the Earth (green circle) orbits, light from the same source must travel different distances—as shown by the dashed, red line. Thus, the light from a star will arrive at different times depending on the earth's position in its orbit.
(b) Plot depicting the time which light arrives at the barycenter versus the Earth. Depending on the location of the Earth in its orbit, the time that an event is observed can shift up to $\pm$ 8.3 minutes with respect to the solar system's barycenter. By imagining the Earth from figure (a) orbiting, and plotting the lengths of the dashed, red line, this sinusoidal plot is obtained.

shown by the diagonal gray line. The dashed, red line shows the added distance that light travels with respect to the solar system's center of gravity—called the barycenter (point b). As the earth orbits the barycenter, this path would grow and shrink. This results in the observation time of an event—such as a transit—shifting slightly throughout the year. The change in the measurement's time is illustrated in the plot shown in Fig. 2.1(b). By visualizing the planet in Fig. 2.1(a) orbiting and graphing the lengths of the dashed, red line through the course of one period, this plot is obtained. The Earth's orbit results in a sinusoidal change in the time of an observed event. This shift could be as much as $\pm$ 8.3 minutes (Carroll & Ostlie 2017). The barycenter, in contrast, is stationary.

Converting to BJD eliminates the sinusoidal effects of Earth's orbit, but this conversion requires a long list of details and many trigonometric calculations. In BJD, the time of an event is as if the measurement took place at the solar system's barycenter (Haswell 2010). This requires measuring the time difference between light's arrival to the observatory on the Earth and the barycenter, and adding this to the time of observation. A simplified equation for converting to BJD is

$$BJD = JD + \frac{\vec{r} \cdot \hat{n}}{c}, \tag{2.1}$$

where JD is the julian date as recorded here on earth, $\vec{r}$ is the vector pointing from the barycenter to the observatory, $\hat{n}$ is the unit vector pointing from Earth to the object, and c is the speed of light (Haswell 2010). Despite the simplicity of Eq. (2.1), the conversion is actually very complex. It requires the exact position of the observatory with respect to the Earth in its orbit at the time of the observations. This requires accurate measurements and extrapolation of the Earth's orbit; the latitude, longitude, and elevation of the observatory taking the data; and the position of the object being observed using right ascension (RA) and declination (dec) coordinates. The calculation then uses numerical trigonometric functions to determine the corresponding three-dimensional angles and vectors from the star to the observatory's position, then to the Earth's center, and then to the barycenter. Through these steps, the time of observation can be adjusted. However, the time standard is not yet in true BJD form.

The full BJD conversion is further complicated by the need to remove relativistic effects caused by Earth's motion. Since Earth is constantly moving—and the barycenter is not—our observed time is slowly retreating from the barycenter's time. This is because a moving clock always runs slower, as shown in relativity. As a result, the velocity and direction of Earth's orbit at the time of observation must also be extrapolated in order to eliminate the relativistic effects of Earth's orbit. Then, an adjustment of leap seconds must be included. Leap seconds are seconds added to account for Earth's clocks constantly running slower. By adding the relativistic effects and the current number of leap seconds—27—the time stamp is fully converted into BJD (Eastman et al. 2010).

Due to these many complications, converting to BJD is almost always accomplished through a lengthy computational computer program. These programs are able to access accurate online databases to obtain the Earth's orbital position and velocity. After providing a list of JD times, the observatory's coordinates, and the star's coordinates, these programs return a list of the converted BJD times. Utilizing a modified form of a Python program from Eastman et al. (2010), this conversion can easily be performed. Now, with the time standards unified, the next step is to remove the noise inherent in our telescope system.

2.1.2 Calibration frames

We perform noise reduction through applying calibration frames to our object images. The calibration images include zero, dark, and flat frames. A zero—or bias—frame is created by measuring the nonzero offset for each pixel on the CCD chip during a zero second exposure. This offset—or bias—is caused by the electronic signal used to read the CCD device. A zero frame generally reads between 1 000 to 2 000 counts—where counts are a measure of how many photons were captured by an individual pixel. By subtracting an average zero frame from each of the remaining frames—dark, flat, and object frames—this offset is removed.

Next, a dark frame is created by measuring the exposure response of the CCD chip for a particular exposure time without any light hitting the CCD chip. An example dark frame is shown in Fig. 2.2(a). The shutter remains closed, yet the CCD chip still reads a signal—shown by the higher response white pixels. The signal appears randomly distributed on the CCD chip, and may be caused by thermal disruptions, dark current, or defects in the CCD chip. Also, this signal increases linearly—in theory—as exposure time is increased. Due to this dependence, dark frames must be matched with flat and object frames of the same exposure lengths. A dark frame generally reads between 1 000 to 20 000 counts. By subtracting an average dark frame of the corresponding exposure time, these sources of random noise are mostly eliminated from our data.

Lastly, a flat frame is an image taken of a uniformly lit field—such as the eastern sky right before sunset. An example flat frame is shown in Fig. 2.2(b). Despite the identical light signal

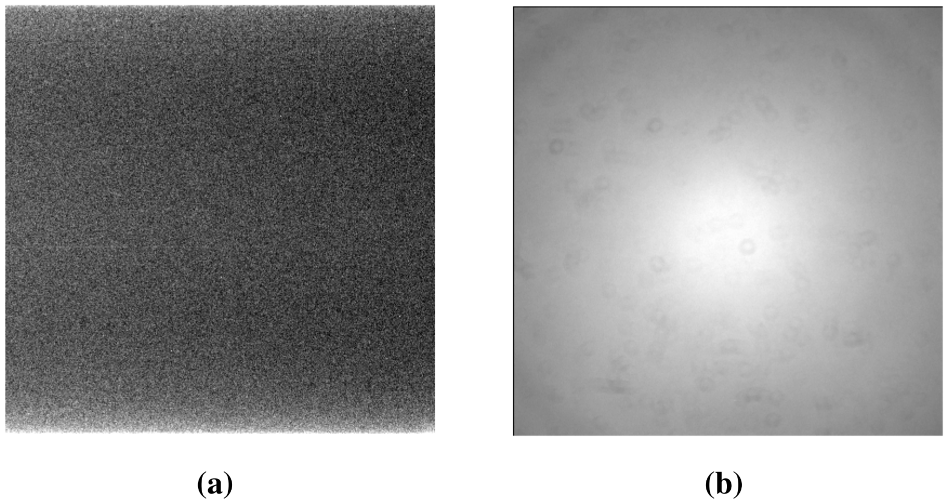


Figure 2.2 (a) An example of a dark calibration frame. The white pixels indicate a higher readout signal while the black pixels indicate lower. Dark frames are mostly random noise, linearly related to exposure time.
(b) An example of a flat calibration frame. The white center indicates a higher response; this effect is caused by the telescope's optics.

from all parts of the field, the CCD chip reads a higher signal (white pixels) in the center of the frame. This result is due to the circular optics used to capture light. This effect is also the most prevalent on object frames, as seen in the unprocessed image in Fig. 2.3(a). A flat frame generally reads between 20 000 to 45 000 counts. To remove this feature, we divide the object pixel values by their corresponding flat frame pixel values. This removes the optical irregularities of our telescope by normalizing the pixel measurements and flattening out the response of the CCD chip. Through applying these three types of calibration frames, nearly all the noise inherent in our telescope system is removed.

Applying calibration frames reduces noise in our system both numerically and visually. After noise reduction processing is completed, we have removed bias offsets, instrumental noise, and normalized the response of the CCD chip. Removing this noise allows the point spread function of the stars captured on the CCD chip to become defined. Calibration frames are taken every time data is taken—in case of unwanted variations in our system—allowing us to remeasure and adjust our

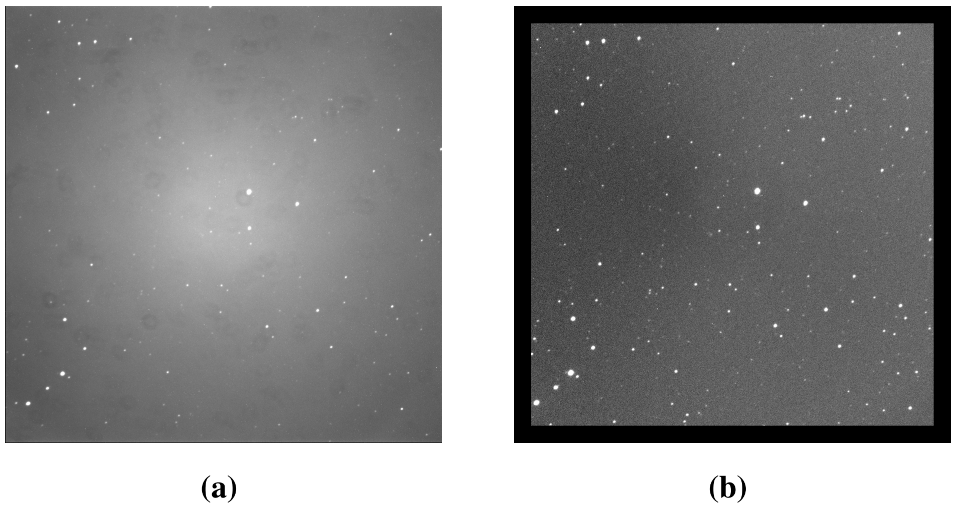
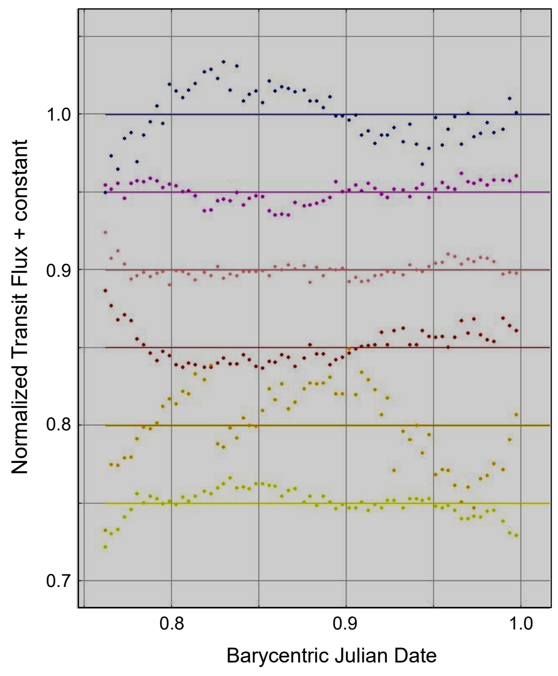


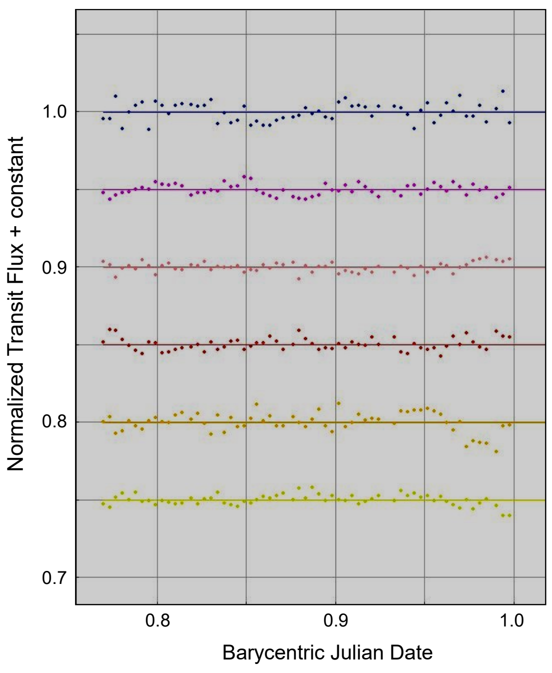
Figure 2.3 (a) Example of a raw image taken with the David Derrick Telescope (DDT) at BYU.
(b) Same image shown after noise reduction processing. Through applying calibration frames—zero, dark, and flat frames—most of the noise inherent in our telescope system is removed.

object frames accordingly. A visual example of an image before and after processing is compared in Fig. 2.3. Before processing—shown in Fig. 2.3(a)—many features of the object frame are obscured by the telescope's noise. As with the flat frames, the center displays a higher response than the corners. After processing—seen in Fig. 2.3(b)—stars on the object frame become clear. Note, we trim the edges of the images while processing due to more nonlinear effects being present in our CCD chip at those points.

The numerical improvements of noise reduction are observed through plotting in AIJ. An example of plots created before and after processing is shown in Fig. 2.4. The six different colored points represent the measured brightness of six different stars in each frame, while the line of the same color represents the best-fit line estimate. Each of the measured stars have been normalized and then shifted in order to see the six individual point plots. These six stars are expected to be stable and unchanging; However, Fig. 2.4(a) shows large variations in the measured brightness of these stars. This is due to the inherent noise in the telescope system. After processing, most of



(a)



(b)

Figure 2.4 (a) Normalized brightness plot for an unprocessed data set. The six different colored points represent the brightness of six different stars being measured. The stars in this plot are predicted to be invariable. However, the inherent noise of the telescope causes large variations in the calculated brightness.
(b) Brightness plot for the same data set after noise reduction processing. Through applying calibration frames, most of the noise inherent in our telescope system is removed. As a result, the six individual stars display the expected consistent output.

this noise is removed, as displayed in Fig. 2.4(b). Once noise reduction has been performed, the measured output from the six stars can be observed as invariable. Through these noise reduction procedures, our observational data is able to be analyzed.

2.2 Overview of Previous Methods

Data preparation procedures are performed using Ohio State's online BJD converter applet (Eastman et al. 2010) and the program Image Reduction and Analysis Facility (IRAF) (Iraf.net Accessed Jan 15, 2019). To convert to BJD, we first use IRAF to replace the observation time with the Julian date (JD). A list of these observation times are then extracted from the object images. This list, the observed star's RA and dec, and the observatory's latitude, longitude, and elevation are input into Ohio State's online applet. The program then performs the needed calculations to convert JD into BJD and returns a list of times in BJD standard. The BJD time is then placed back into the corresponding object frame using IRAF. This completes the required conversion to BJD time.

Noise reduction processing requires many procedures. Written in a combination of FORTRAN and C, IRAF generally uses a command-line structure for calling and executing tasks. This means that each command is manually typed into the program. To complete the noise reduction procedures, first, all images are trimmed—as discussed for Fig. 2.3(b). Next, a master zero is created by averaging all zero calibration frames. This master zero is then subtracted from each of the remaining frames—dark, flat, and object. Next, dark frames are averaged and a master dark is generated for each exposure time. The master darks are then applied to the flat and object frames of corresponding exposure times. Lastly, flat frames are averaged to create a master flat for each filter used in the object frames. These master flats are then divided into the object frames of corresponding filter. Each of these steps for creating and applying master calibration frames requires a large list of commands to be typed into IRAF.

Manual reduction can become extremely tedious. On an average night, there could be two objects observed—each with a unique exposure time and filter—and between three to seven different exposure times used to obtain flat calibration frames. In applying dark frames, this would require up to nine different master dark frames to be created. Then, each master dark would need to be applied to each of the unique nine image sets. However, it becomes tedious to create and apply a master dark for each of these individual frames. Often—in an effort to save time—an average valued master dark is applied to all of the flat frames instead. This simplifies the manual reduction process, but it also may remove less of the inherent noise.

Despite the reduction process being similar for every data set, each research assistant processes images slightly different. Some may choose to implement additional steps while others employ shortcuts. There are multiple command types which can be used to complete the same task in a unique way. This can unintentionally add variations to the processed data, changing the results somewhat. Otherwise, the parameters used in many of the commands can be adjusted. This allows for flexibility in the processing steps, though it also creates further possibilities for deviations in the processed images. In addition, human errors can be unintentionally introduced into the processed images. If a command was incorrectly entered or forgotten, the analysis results can be noticeably impacted. Depending on the student processing the images, the command-line results can create strikingly different results. Instead, using an automated program would remove variations, human errors, and would standardize processing procedures.

Image Reduction and Analysis Facility (IRAF) has a programming mode, which uses C-style script files to execute tasks. By calling these files, IRAF will iterate through each line, allowing commands to be automated. Recently, a student chose to create some script files in order to aid the reduction process (Patterson 2017). This program, however, came with a few issues. First, there were still many steps left to be completed by the manual command-line IRAF functions, including the process of converting the image's time stamps into BJD. Also, the program was unable to work with multiple targets. If a night included observing more than one target, each object needed to be placed in its own file with all of the needed calibration frames. Lastly, this program was unable to differentiate between the exposure times. This had a large impact on applying dark calibration frames. As a result, only one dark frame exposure time is used—the one that matches the object's exposure time.

Since the signal is roughly linear, a large exposure time causes the dark frame pixel counts to grow large as well. While this is the desired effect for the object frames—taken with the same exposure time—it is not beneficial when applying long exposure dark frames to short flat frames. A general exposure time for an object frame is between 60 to 120 seconds, while a normal flat frame has an exposure between 1 to 5 seconds. When subtracting this high exposure master dark off of the flat frames, unusually small pixel values become plausible. When dividing the flat frames into the object frames, these values could add more noise into the system by not fully flattening out the response of the CCD chip. Despite these weaknesses, this program has been a helpful tool when required to quickly process data. It also established a precedence for further research into the programming mode of IRAF.

2.3 The Pipeline

Using scripts written in C, I created a program to automatically process images. We call this program the pipeline, since it utilizes a linear sequence of logic commands to transform the images from raw to processed data. This program was developed utilizing a number of online resources (Anderson 1989; Iraf.net Accessed Jan 15, 2019; Shames & Tody 1986) and it utilizes more programming capabilities of IRAF to logically investigate a night's set of images. The pipeline doesn't have the weaknesses of its predecessor: it automates the conversion to Barycentric Julian Date, can handle any number of targets observed, and the pipeline can create and apply master dark frames to the flat

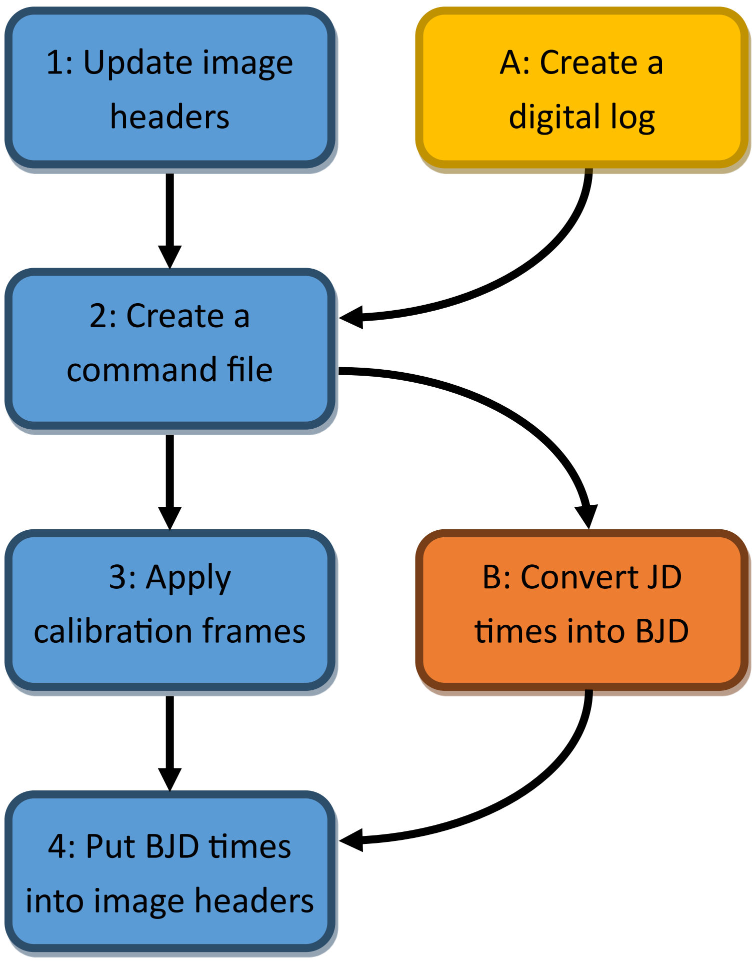

1: Update image headers
A: Create a digital log
2: Create a command file
3: Apply calibration frames
B: Convert JD times into BJD
4: Put BJD times into image headers

Figure 2.5 Flow chart outlining the steps of the pipeline program. The yellow box (step A) is a record made manually by the researchers during observations that includes required information about the star being observed. The orange box (step B) denotes a section of Python code—modified from Eastman et al. (2010)—used for the time conversion. The blue boxes (steps 1-4) are the linear procedures performed by the pipeline automatically in IRAF. These four steps, along with the two additional procedures, perform all of the noise reduction procedures.

and object frames of corresponding exposure times.

The pipeline performs the following steps, as outlined visually in Fig. 2.5. Step 1: the pipeline uses IRAF commands to update all of the images' headers. This tells IRAF the image types and filters—which is required when applying calibration frames. Step 2: the pipeline creates a command file of the extension CMDS. However, this file requires additional information that can't be found in the images themselves. As a result—while using the telescope—researchers create a digital log of their observations. This step is found in Fig. 2.5 as step A. The log includes information about the object's right ascension and declination—required for the CMDS file—as well as information about the observations performed. With the digital log and a command file created, the pipeline is then able to calculate the Julian Date (JD) of the observations.

Next, the pipeline is split into two consecutive steps. Step 3: the pipeline applies calibration frames. Using a variety of logic structures, the program sorts through all of the raw images and only performs noise reduction on the required images—based on which calibration images are needed to process the object frames from that night's observations. The program applies zero, dark, and flat calibration frames in order as explained in section 2.1.2 and section 2.2. This portion of the program requires the most time in order to complete all of the needed steps. As a result, researchers are able to simultaneously run a quick Python script at this time, which is step B in Fig. 2.5. Using a modified form of René Tronsgaard's module, this Python program converts the JD times from step 2 into Barycentric Julian Date (BJD) by making a query to Ohio State's online applet automatically (Eastman et al. 2010). Once applying calibration frames in completed, the pipeline places the BJD times into their respective image headers—step 4. After completion of these six steps, the images have been processed and are ready for analysis.

Unlike its predecessor, the pipeline is able to focus on only the data needing processing, allowing it to run quickly while retaining all calibration frames. The pipeline is able to sort through the night's data automatically. Through many logic structures, it can constrain its steps based on the filters and exposure times of the object and flat frames. The pipeline then responds accordingly, processing only the required calibration and object frames. This allows the pipeline to process each flat frame only with its corresponding dark frame. Since it is run through low-level language script files—with most of its tasks being logic statements—the pipeline is able to quickly accomplish all data processing tasks.

To avoid potential issues for future students—due to updates or changes to IRAF—I added detailed explanations of each portion of this program. This is perhaps the most essential task I performed while creating the pipeline. I added many helpful—though concise—comments and error checking statements throughout the pipeline. I also listed the current version of IRAF and Ubuntu—our current operating system—used when the pipeline was developed. Lastly, I added print statements that allows a student to troubleshoot the pipeline step-by-step, should there ever be a major concern. All these tasks were performed for two reasons. First, so this program could be easily adjusted and updated to fit new equipment, new projects, or new systems. Second, one day—inevitably—the pipeline will experience a major technical issue. This may be due to an update in IRAF or Ubuntu, or perhaps due to changes made in the observational data sets. Regardless of the concern, I hoped to create a readable program that could be understood by a student with much less IRAF experience and that can be altered in the event of an issue. The pipeline is constructed so that if it were stopped—due to an error—almost any research student should be able to locate where that error occurred. After reading the output print statements and error codes, they should understand how to resolve that error. This addition makes the pipeline reliable, durable, and more permanent.

2.4 Discussion of challenges encountered

Besides creating a program that could be understood by almost any research student, the largest challenge met while creating the pipeline was finding coding practices in IRAF's programming mode. There are distinct differences between the command-line and programming modes of IRAF. Most use IRAF for the command-line abilities, so documentation on the programming mode is less organized and less abundant. The syntax used in programming mode is different from the command-line, meaning many functions and programs would not work in both modes. In general, the most helpful resources—Shames & Tody (1986) and Anderson (1989)—were not recent resources. As a result, much of the documentation about IRAF's programming mode was out of date and many expressions and functions that I tested were no longer recognized in IRAF—having been removed by a more recent update. In order to find usable functions, I was required to perform a considerable amount of research followed by experiments to test my findings. I often began script files with only a goal in mind, then I would research and experiment until I found functions that could help me achieve that goal. This resulted in building script files, piece by piece, and updating these files many times as I discovered more about IRAF's programming mode. I often rewrote entire files upon finding additional functions that met my original goal. Despite these challenges, I was able to complete an effective, free, automated program that performs all of the tasks we require.

Chapter 3

Results and Conclusions

3.1 Results

The pipeline performs noise reduction procedures more effectively than the average student researcher. Over the course of a few semesters, students have recorded their reduction times whenever they performed noise reduction manually. Once the pipeline was completed, I reprocessed those same raw images with it and recorded the time required by the pipeline for processing. A complete list of the number of images in each data set as well as the reduction times—manual and automatic—can be found in Table A.1 in Appendix A. In Table 3.1, the averages and standard deviation are listed for this sample for the number of images, manual reduction time, and the pipeline's reduction time.

The pipeline shows many advantages over manual reduction. First, from Table 3.1, the average reduction time for the pipeline was significantly less than the manual time. The pipeline is able to perform the noise reduction tasks quickly and more efficiently than our students. Also, from Table 3.1, the standard deviation of the pipeline's reduction is considerably lower, meaning the pipeline is able to process data sets of differing sizes in approximately the same amount of time.

Table 3.1 Average times for manual and automated noise reduction are compared for a sample of data sets. The pipeline was able to perform noise reduction procedures quicker on average.

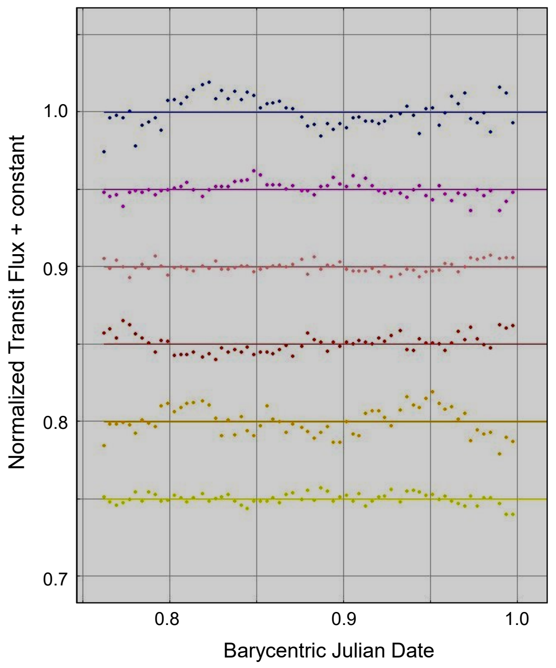
	Number of images	Manual Time (min)	Pipeline Time (min)
$\bar{x}$ [a]	198	24.9	9.1
s_x [b]	71	9.1	3.1

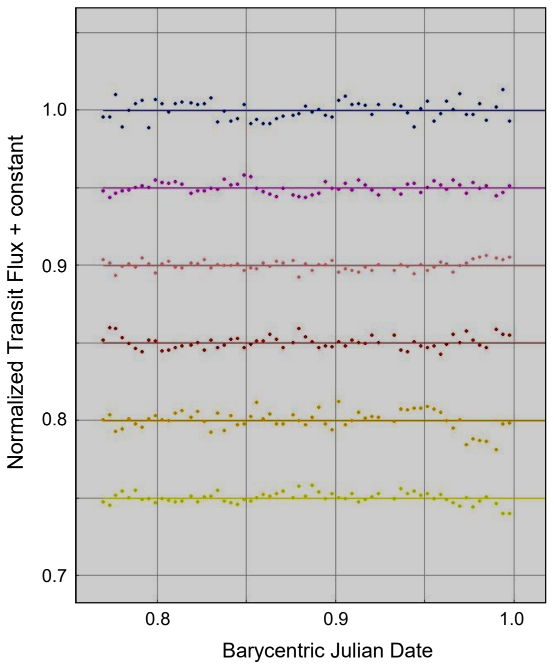
[a] Average value

[b] Standard deviation of sample

This shows that the pipeline is more uniform in its processing than the manual procedures. The differences in how each student uniquely reduces data becomes clear in the wider spread of time required for manual processing. The pipeline, on the other hand, performs the exact same tasks in the same order with the same, well-proven parameters. This creates a uniform data set to be used in analysis.

Additionally, the pipeline prevents human errors. These errors—though they are rare—are always possible in the manual process. Steps may be forgotten or settings may be changed inadvertently. An example of such a mistake is compared in Fig. 3.1—using the same data set from Fig. 2.4. In Fig. 3.1(a), only one short exposure time of dark frames was used in processing. The master dark created from these short exposure dark frames matched the flat frames' exposure times but was significantly shorter than the object frames' exposure length. As a result, variations in the stars' points are still observed. Comparing to the unprocessed data in Fig. 2.4(a), this inaccurate reduction still removes a significant portion of the inherent noise. However, compared with the pipeline's reduction in Fig. 3.1(b), a simple mistake results in noticeable variations between these two processed data sets. The pipeline—on the other hand—consistently performs each step. It uses the same procedures and settings, removing possible fluctuations and mistakes introduced by





(a) **(b)**

Figure 3.1 (a) Brightness plot for a processed data set with a human error, using the same data set from Fig. 2.4. In processing these images, only one exposure time of dark frames was used. The master dark frame matched the short exposure flat frames, but was too short for the object frames. As a result, noise remains in the calculated brightness of these six stars.
(b) Brightness plot for the same data set after noise reduction processing by the pipeline. The pipeline removes more of the noise inherent in our telescope system, so the six stars display the expected invariable output.

manually processing the data.

Lastly, the pipeline creates cleaner data sets. The pipeline is able to remove more inherent noise than most students performing manual procedures. This is especially true when a student takes a shortcut or makes an error, as compared in Fig. 3.1. When comparing correctly and carefully processed images—especially for an experienced research student—the resulting light curves still differ, though only by small fluctuations. The discrepancies are difficult to observe, unlike in Fig. 3.1. However, the cleaner data—as reduced by the pipeline—will include less random variations and should be closer to the true solution. By using the pipeline, it is more likely that the analysis will find the correct exoplanet result.

3.2 Conclusions

In order to eliminate the obstacle of noise reduction processing, an automated program was developed. The pipeline is able to perform all tasks needed for processing, removing the challenges of manual reduction. As was shown in Section 3.1, the pipeline has four main advantages over manual reduction: the time required is reduced, the resulting data sets are more uniform, human errors are avoided, and the processed data sets are cleaner. Through use of the pipeline, more research time can be focused on analyzing the data. Findings are returned to KELT sooner, allowing possible exoplanet candidates to be categorized faster. Lastly—and most notably—the pipeline allows more freedom for research time. Through the pipeline, students don't need to spend their limited research time manually performing the noise reduction procedures. As a result, they can instead devote more time to analyzing results and working on other research projects. The pipeline has improved our research potential by removing barriers in our research process, improving our processed measurements, providing a way to return findings quicker to KELT, and allowing students to devote more time to diverse projects.

3.3 Directions for further work

Upon completing such a beneficial program, the obvious next step would be to offer it freely. Noise reduction is a common obstacle for all Astronomy research, not just exoplanet studies. As a result, the pipeline could be equally beneficial to any astronomy program—especially since it is all performed using free, publicly available software. At current, the pipeline contains procedures specifically required by the KELT team. This would make it an excellent program for all members of the KELT follow-up network to use in their exoplanet research. With a few minor adaptations, the pipeline could be applied to many more research topics. As a result, our next objective is to post this program to an online repository—such as GitHub—to make our pipeline publicly available. Then—through presentations, posters, publications, and the KELT follow-up network—we will share the advantages and possibilities of this program. Consequently, we can contribute to accelerating Astronomy research by removing the obstacle of noise reduction.

Appendix A

Noise Reduction Time Tests

Table A.1 Full listing of all data sets tested, comparing the pipeline's time against a random research student performing manual noise reduction procedures. The Pipeline is consistently quicker than the average student researcher.

Dataset number	Number of images	Manual Time (min)	Program Time (min)
1	115	5.4	5.01
2	200	7.3	8.5
3	240	13.0	9.4
4	115	13.2	5.2
5	240	13.9	9.4
6	217	16.8	10.2
7	90	20.5	5.2
8	182	21.0	8.4
9	159	21.0	7.3
10	130	24.0	6.2
11	90	25.2	6.1

Dataset	Number of images	Manual Time (min)	Program Time (min)
12	120	27.0	5.8
13	290	27.6	13.9
14	360	28.2	14.0
15	254	30.0	15.5
16	200	30.0	8.5
17	282	30.0	13.2
18	290	30.0	13.9
19	217	31.2	10.2
20	180	33.0	8.0
21	194	33.0	8.4
22	165	40.2	7.3
23	231	52.2	10.0

Bibliography

Anderson, E. 1989, An Introductory User's Guide to IRAF Scripts (Central Computer Services)

Carroll, B., & Ostlie, D. 2017, An introduction to modern astrophysics, 2nd edn. (New York City, NY: Cambridge University Press)

Collins, K., et al. 2018, Astronomical Journal, 156

Croswell, K. 1997, Planet quest, 1st edn. (New York City, NY: The Free Press)

Eastman, J., Siverd, R., & Gaudi, S. 2010, Astronomical Society of the Pacific, 122

Haswell, C. 2010, Transiting exoplanets, 1st edn. (New York City, NY: Cambridge University Press)

Iraf.net. Accessed Jan 15, 2019, https://iraf.net

Kielkopf, J. Accessed Apr 9, 2019, https://www.astro.louisville.edu/software/astroimagej/

Oberst, T., et al. 2017, Astronomical Journal, 153

Patterson, A. 2017, Thesis Archive, Brigham Young University, Provo, UT.

Pepper, J., Gaudi, S., & Stassun, K. Accessed Jan 15, 2019, https://keltsurvey.org/

Pepper, J., et al. 2007, PASP, 119

Shames, P., & Tody, D. 1986, A User's Introduction to the IRAF Command Language (NOAO)

Index