To what Extent can Using SKA-Mid Improve Time Delay Estimations of Lensed Objects and Ultimately the Hubble Constant?

Gabriel Dominic Richard

November 2025

Abstract

The quad-image gravitational lens system B1608+656 was used as a basis to simulate time delays, with observations performed using the SKA, AA4 configuration. The system was observed over 365 days. The data from these observations were sampled at different cadences, from which light curves where then constructed to obtain the time delays between the objects. With respect to data previously obtain from a variety of different lensing systems, the SKA showed an improved performance in the uncertainty of the measured time delays, with uncertainties brought down from days to hours. The implication of this on cosmology is that the Hubble constant, H_0, may be more accurately determined as the SKA has improved precision over previous generation interferometers.

1 Introduction

1.1 Cosmology and The Hubble Constant

Cosmology is a branch of science dedicated to determining the origins of our Universe, it's evolution, and it's eventual fate. In addition, cosmology concerns itself with the structure and contents of the Universe [1]. These contents include radiation and matter, along with dark matter and dark energy, all of which have a role to play in the behaviour of the Universe. It is easy to see how the Universe is structured on a local scale, but what about on a larger scale? The cosmological principle states that the Universe is homogeneous and isotropic, meaning that when viewed at large, the Universe looks the same from every point and the same in every direction [2]. This principle then formed the basis of the Friedmann–Lemître–Robertson–Walker (FLRW) concordance model.

Two important constructs were required to define the above model: Einstein's field equations and an appropriate metric. A metric is a geometrical description of space-time [1], where Einstein's field equations dictate the response of the metric in the presence of matter [3]. The metric used in defining the above model is the Robertson-Walker (RW) metric, which is consistent with the idea that the Universe homogenous and isotropic [4]. The RW metric consists of two terms called the scale factor a and the curvature parameter k. The curvature parameter defines the geometrical structure of the Universe[1] and the scale factor describes the rate of expansion [2] (this will be discussed below).

Friedmann then used Einstein's field equations and the RW metric to derive what are know as the Friedmann equations. Through these equations, it was found that the Universe could either expand or contract. This was an important result, as at the time, prominent figures such as Einstein believed the Universe to be static[2] [4]. Edwin Hubble was the first person to confirm of this expansion through the observation of distant galaxies and their associated redshifts [5]. Thus, in honour of Edwin Hubble, the present day expansion rate of the Universe was termed the Hubble constant and denoted H_0.

There are two Friedman equations of significance to the discussion at hand. The first equation descibes the expansion rate of the Universe while the other descibes the acceleration of this expansion. These equations depend on the physical composition of the Universe. They require it's pressure , the associated energy density (contributions from matter, radiation and dark energy) and the cosmological constant [4].

[1] These geometries are spherical, hyperbolic and flat.

[2] This is what led Einstein to adding the cosmological constant Λ to his field equations

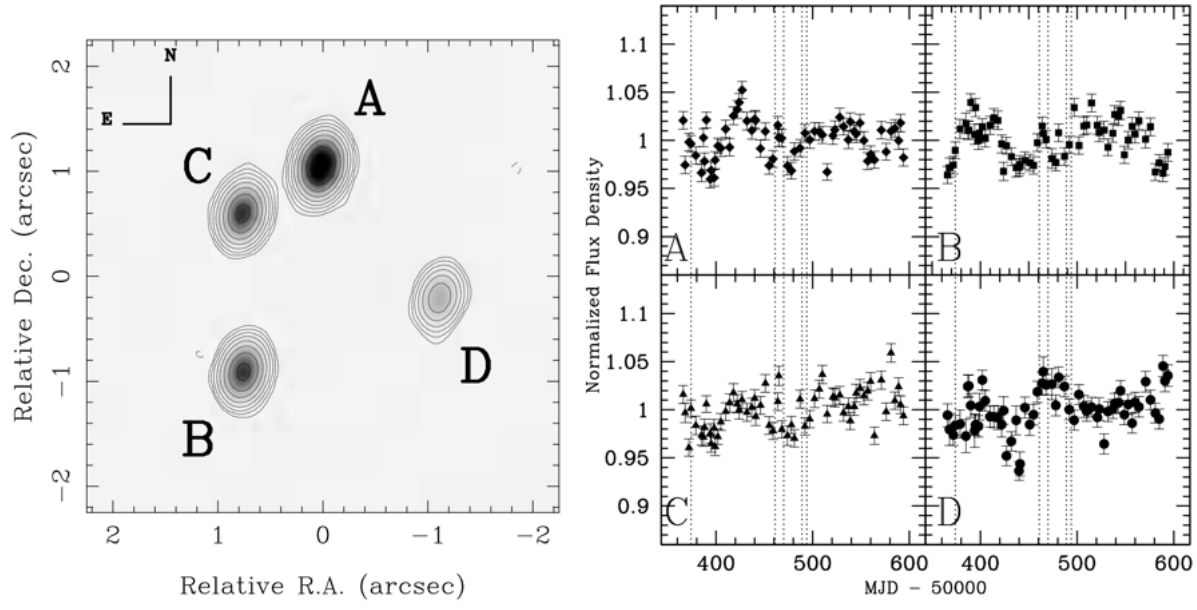
But more particularly, the Friedmann equations described are time dependant. Thus, they are used to predict not only the current, but both the future and past expansion of the Universe.

Thus, the FLRW model is a model that assumes a homogenous and isotropic Universe that is non-static, with the possibility of expansion or contraction. However, it is widely accepted that the Universe is expanding. But the rate at which it expands has been a point of contention, and so contentious in fact that it was termed the Hubble tension. The Hubble tension arises due to differing values of the Hubble constant based on the measuring technique, two prominent methods of measurement being through the local distance ladder and the cosmic microwave background (CMB) [6]. The CMB method relies on using properties such as matter densities and acoustic peaks to determine the Hubble constant, where the local distance ladder relies on the relationship between the distance of the object and it's redshift[3] [6].

There exist other methods in which time delays can be determined. One method of particular interest involves gravitationally lensing an object and measuring the time delays associated with the images produced by the lens [7].

1.2 Gravitational Lenses and Time Delays

A gravitational lens may be described as a massive object in the foreground which distorts and magnifies the light of an object in the background [8]. As mentioned previously, Einstein's field equations dictate the behaviour of space-time in the presence of mass. In the presence of mass, space-time is distorted, with any light rays in the vicinity of the mass following a geodesic[4] [3]. Simply put, a geodesic is the shortest possible path between two points in curved space-time [9]. In the absence of any mass, the path a light ray will travel is a straight line. Hence, after interacting with a mass, the light ray will have been deflected off it's initial trajectory. This deflection is quantified through a variable called the deflection angle.



(a) B1608+656 Configuration

(b) B1608+656 Object Light Curves

Figure 1: To the left is a gravitationally lensed image containing four objects, each of varying brightness. To the right are the normalised flux densities of the respective objects over a season (Sourced from [10]).

There are three type of gravitational lensing: weak lensing, strong lensing and microlensing [11]. Weak lensing corresponds to weak distrotions and minimal magnification [12]. Many slight displacements and distortions from observed sources come as a result of weak gravitational lensing [11]. This type of lensing does not produce multiple images. Strong lensing corresponds to strong deflection which produces multiple images of a background source as a result of a higher mass density in the foreground object [11]. Often times these foreground objects are large galaxies or galaxy clusters. Microlensing is similar to

[3] The object is often a cepheid variable or supernova type Ia [6].

[4] More precisely, the null geodesic

strong lensing in the sense that multiple images are produced, however, the angular separation of the images is too small for the them to be resolved. The objects responsible for this type of lensing are concentrated in size, and generally have masses up to a maximum order of 10^6 solar masses [12].

The lensing type of most interest is strong gravitational lensing. As stated in the previous paragraph, strong lensing produces multiple images. To note, the images produced by a lens depends highly on the mass distribution and the resulting gravitational potential of the lens. This brings to light an important concept: from the observers perspective, gravitational lensing depends only on the two dimensional projection of the mass distribution of lens [13]. Any change in the mass distribution of the lens that is along the line on of the observer will not result in a change of the produced image. However, if either foreground lens moves or the background object moves, the resulting image may change. A varying projected mass distribution will also result in a changing image.

Figure 1(a) shows the quad lens system B1608+656 from which four images have been produced. Lenses that produce multiple images that are well separated, such as in the figure above, are particularly convenient with regards to obtaining time delays. Firstly, time delays are the result of the gravitation potential of the lensing mass [11]. Based on the position on the lens and object with respect to observer, each light ray will experience a different gravitational potential, and ultimately, a different path length. The arrival time of the light from each object will then vary, and it is this difference in arrival time that constitutes a time delay. With well resolved objects, the flux of each can me measured and from this, a light curve may be constructed. It is these light curves that reveal the intrinsic variability of the lensed object, and if this is known, the light curves will provided the means through which the time delays are measured. Figure 1(b) displays the normalised light curve of each object in Figure 1(a).

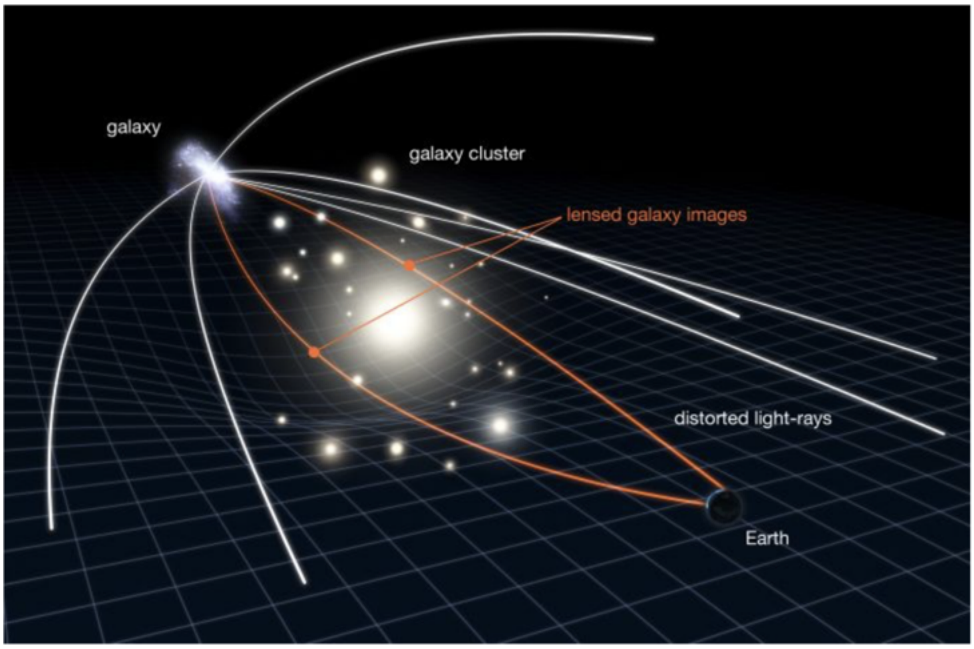


Figure 2: Visualisation of gravitational lensing by a galaxy cluster (sourced from [14]).

Figure 2 showcases how the light rays from the source galaxy[5] are deflected by the lensing galaxy cluster. Some rays traverse a greater path to Earth while others travel shorter paths. It's also quite evident that many of the deflected light rays do not deflect enough, and travel past Earth. This brings to light an important point: if the lensing galaxy cluster was not there, much of the light that was deflected would then travel in a straight line, missing Earth. Light is the primary form of information gathering from distant sources within the Universe, and so, the gravitational lenses are a great aid in relaying extra information to the observer. The more information being received, the better the collected data is. In the case of the time delays, the more light received from each object, the stronger the data recordings are and the better the reconstruction of the intrinsic light curve of the source object is.

Through the use of gravitational lensing and the right observational equipment, the gravitationally lensed

[5] The image uses a galaxy, however the object could be a quasar or any other source of interest.

time delays may be determined. The choice of equipment in this instance is an interferometer, and more specifically, the use of the Square Kilometer Array (SKA) is of particular focus. The SKA is a next-generation interferometer that is anticipated to provide a better sensitivity and resolution than previous interferometers [15]. This improved sensitivity will not only allow for better observations of fainter objects, but will result in more accurate data. Coupling this with better resolving power, the SKA will serve as a tool to collect more precise data with regards to time delays of lensed objects.

1.3 Aim

The aim of this thesis is to determine the extent through which the SKA can accurately record the flux density of the objects formed by a point source which has undergone strong gravitational lensing. The flux density of each object then determines how accurately the light curve reconstructions of the objects are, and in effect, the determined time delays.

This was done through generating light curves of gravitationally lensed images, and simulating observations using the SKA antenna configuration. Theses simulated observations were then used to construct the resulting light curves, which were then used in determining the time delays.

2 Mathematical Background

2.1 The Friedmann Equations

The two Friedmann equations mentioned in Section 1 are [1, 2]

$$H^2(t) = \left(\frac{\dot{a}}{a}\right)^2 = \frac{8\pi G\rho}{3} + \frac{\Lambda}{3} - \frac{k}{a^2} \tag{1}$$

and

$$\frac{\ddot{a}}{a} = \frac{\Lambda}{3} - \frac{4\pi G}{3}(\rho + \frac{3p}{c^2}). \tag{2}$$

Equation (1) descibes expansion rate of the Universe at a given time t, and equation (2) describes the acceleration rate of the universe. The parameters are as follows: G is the gravitational constant, Λ is the cosmological constant, ρ is the density of the universe, p is the universal pressure, a is the scale factor and k is the curvature parameter. H is referred to as the Hubble parameter. If t_0 denotes current time, then $H(t_0) = H_0$.

2.2 Gravitational Lensing and Time Delays

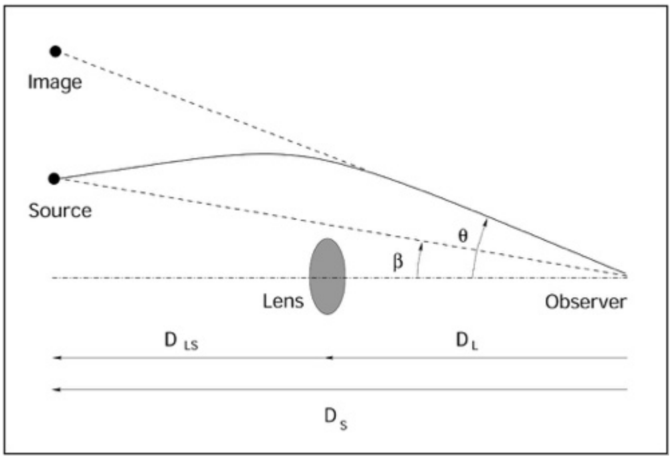


Figure 3: Lensing Diagram (Sourced from [16]).

Figure 3 proves a good basis on which to describe the geometry of the setup. D_{L} is the distance between the observer and lens, D_{LS} is the angular distance between the lens and the source while finally, D_{S} is the angular distance between the observer and the source. β describes the true angular position of the source while θ describes it's perceived image angular position. The equations below are based on the thin-lens approximation, which approximates the lens to be thin if distance in which the majority of deflection takes place is much smaller than D_{L} and D_{LS} [13]. With this approximation, the projected mass distribution lies within a plane called the lens plane, and similarly, the source lies within a plane referred to as the source plane [12]. These planes lie parallel to one another, and are perpendicular to a line intersecting them referred to as the optical axis [12].

The trajectory of a light ray may be represent through the coordinates [12]

$$\vec{r} = (\xi_1(\lambda), \xi_2(\lambda), z(\lambda)) \,, \tag{3}$$

where ξ_i denotes impact parameter[6] of the given dimension, and z is the direction along which the light ray propagates [12]. λ in this instance is an affine parameter. Affine parameters parametrise geodesic curves [3], which is what the light ray will follow in the presence of matter. The coordinates ξ_1 and ξ_2 lie in the lens plane, measured relative to the intersection of the optical axis with the lens plane. Thus, points on the lensing plane may be described through the impact vector $\vec{\xi}$, where [12]

$$\vec{\xi} = (\xi_1, \xi_2) \,. \tag{4}$$

If the lens is not a point mass, the mass density of the lens may be described through $\rho(\xi_1, \xi_2, z)$. With the current notation, it is convenient to introduce a quantity referred to as the surface mass density, given by [13]

$$\Sigma(\vec{\xi}) = \int \mathrm{d}z \, \rho(\xi_1, \xi_2, z). \tag{5}$$

Recall that the lensing plane contains a projection of the mass distribution. What the surface mass density describes is the mass distribution projected onto the lensing plane [13]. As evident, the surface mass density is a scalar field, accepting point's on the lensing plane and in return, outputting the mass density of that particular point on the plane.

Accordingly, an infinitesimal mass element of the lens is given by $\mathrm{d}m = \rho \mathrm{d}V$, whose position and impact vector is given by [12]

$$\vec{r}' = (\xi_1', \xi_2', z') \quad \text{and} \quad \vec{\xi}' = (\xi_1, \xi_2) \tag{6}$$

respectively. Thus, the impact vector of the light ray relative to the infinitesimal mass element is give by

$$\vec{R} = \vec{\xi} - \vec{\xi}' \,. \tag{7}$$

With all the necessary components described above, the deflection angle due to a mass density described by Σ is thus given by [12]

$$\hat{\alpha}(\vec{\xi}) = \frac{4G}{c^2} \int d^2\xi' \, \frac{\vec{\xi} - \vec{\xi}'}{|\vec{\xi} - \vec{\xi}'|^2} \Sigma(\vec{\xi}') \,. \tag{8}$$

Note that $\hat{\alpha}$ is a two dimension quantity.

Similarly to how $\vec{\xi}$ was defined on the lens plane, $\vec{\eta}$ defines points on the source plane, with

$$\vec{\eta} = (\eta_1, \eta_2) \tag{9}$$

and utilising the geometry of the setup, $\vec{\eta}$ may be defined as [12]

$$\vec{\eta} = \frac{D_\mathrm{S}}{D_\mathrm{L}} \vec{\xi} - D_\mathrm{LS} \hat{\alpha}(\vec{\xi}) \,. \tag{10}$$

Again, exploiting the geometry of the setup, and using small the angle approximation, $\vec{\eta}$ and $\vec{\xi}$ may be expressed as

$$\vec{\eta} = D_\mathrm{s} \vec{\beta} \quad \text{and} \quad \vec{\xi} = D_\mathrm{d} \vec{\theta} \tag{11}$$

and then substituting this into equation (10) yields [12]

$$\vec{\beta} = \vec{\theta} - \vec{\alpha} \,, \tag{12}$$

with

$$\alpha(\vec{\theta}) = \frac{D_\mathrm{LS}}{D_S} \hat{\alpha}(D_\mathrm{L} \vec{\theta}) \,, \tag{13}$$

[6] The impact parameter is the perpendicular distance from the centre of mass.

where $\vec{\alpha}$ is referred to as the scaled deflection angle. Equation (12) is referred to as the lens equation.

Equation (12) expresses the true angular position of the source through it's apparent angular position, where $\vec{\alpha}$ is a function of $\vec{\theta}$. Equation (12) is non-linear [13], implying that multiple $\vec{\theta}$'s may correspond to a singular $\vec{\beta}$. Physically, the implication is that for a single source, there may be multiple images as a product of the lensing.

Looking back to equation (8) and using equation (11)

$$\vec{\alpha}(\vec{\theta}) = \frac{D_{\mathrm{LS}}}{D_{\mathrm{S}}}\frac{4G}{c^2}\int d^2\theta' D_{\mathrm{L}}^2\,\frac{D_{\mathrm{L}}(\vec{\theta}-\vec{\theta}')}{D_{\mathrm{L}}^2|\vec{\theta}-\vec{\theta}'|^2}\Sigma(D_{\mathrm{L}}\vec{\theta}') = \int d^2\theta'\,\frac{(\vec{\theta}-\vec{\theta}')}{|\vec{\theta}-\vec{\theta}'|^2}\left(\frac{D_{\mathrm{L}}D_{\mathrm{LS}}}{D_{\mathrm{S}}}\frac{4G}{c^2}\Sigma(D_{\mathrm{L}}\vec{\theta}')\right) \tag{14}$$

with [12]

$$\kappa(\vec{\theta}) = \frac{\Sigma(D_{\mathrm{L}}\vec{\theta})}{\Sigma_{\mathrm{crit}}} \quad \text{and} \quad \Sigma_{\mathrm{crit}} = \frac{D_{\mathrm{S}}}{D_{\mathrm{L}}D_{\mathrm{LS}}}\frac{c^2}{4\pi G}\ . \tag{15}$$

The scaled deflection angle may be written as

$$\vec{\alpha}(\vec{\theta}) = \frac{1}{\pi}\int d^2\theta'\,\kappa(\vec{\theta}')\,\frac{\vec{\theta}-\vec{\theta}'}{|\vec{\theta}-\vec{\theta}'|^2}\,, \tag{16}$$

where κ is referred to as the convergence[7] while Σ_{crit} is referred to as the critical surface mass density. The critical surface mass density is the criteria that defines the type of lensing that occurs [12, 13]. For $\Sigma > \Sigma_{\mathrm{crit}}$, strong lensing occurs and multiple images are formed.

Using the identity $\nabla_\theta \ln|\vec{\theta}| = \vec{\theta}/|\vec{\theta}|^2$, the deflection angle can be written as the gradient of the deflection potential [12, 13]

$$\vec{\alpha}(\vec{\theta}) = \frac{1}{\pi}\int d^2\theta'\,\kappa(\vec{\theta}')\,\nabla_\theta \ln|\vec{\theta}-\vec{\theta}'| = \nabla_\theta\left(\frac{1}{\pi}\int d^2\theta'\,\kappa(\vec{\theta}')\,\ln|\vec{\theta}-\vec{\theta}'|\right) = \nabla_\theta\psi(\vec{\theta}) \tag{17}$$

where

$$\psi(\vec{\theta}) = \frac{1}{\pi}\int d^2\theta'\,\kappa(\vec{\theta}')\,\ln|\vec{\theta}-\vec{\theta}'| \tag{18}$$

with ψ the deflection potential. The deflection potential models the deflection characteristics of the projected mass distribution. The above result now allows for the formulation of the time delays defined through the ψ, $\vec{\beta}$, and $\vec{\alpha}$. This is done by considering the Fermat potential. Firstly, Fermat's principle states that light will travel the path requiring the shortest time to traverse. Fermat's potential is then given by [12]

$$\tau(\vec{\theta},\vec{\beta}) = \frac{1}{2}(\vec{\theta}-\vec{\beta})^2 - \psi(\vec{\theta}). \tag{19}$$

Fermat's potential describes the time required for a light ray to travel from the source to an observer, through a gravitational lens [12]. The concept of a time delay only becomes applicable when comparing two or more light rays from the same source. Hence, for an image at $\vec{\theta}_{\mathrm{A}}$ and an image at $\vec{\theta}_{\mathrm{B}}$, with the source object at $\vec{\beta}$, the time delay Δt_{AB} between image light rays originating from A and B is given by [7]

$$\Delta t_{\mathrm{AB}} = \frac{D_{\Delta t}}{c}\left[\tau(\vec{\theta}_{\mathrm{A}},\vec{\beta}) - \tau(\vec{\theta}_{\mathrm{B}},\vec{\beta})\right] \tag{20}$$

with $D_{\Delta t}$ referred to as the time-delay distance and given by

$$D_{\Delta t} = (1+z_L)\frac{D_{\mathrm{L}}D_{\mathrm{S}}}{D_{\mathrm{LS}}}\,. \tag{21}$$

Here, z_L is the redshift of the lensing object (a galaxy in this instance). The time-delay distance is how the time delays are related to the Hubble constant. The Hubble constant is inversely proportional to the time-delay distance [7] and formulated as

$$H_0 \propto D_{\Delta t}^{-1}. \tag{22}$$

[7] κ is also referred to as the dimensionless surface mass density.

2.3 Interferometry

2.3.1 Brief Mathematical Review

Of all the important mathematical concepts in interferometry, there are two of particular interest: the Fourier transform and convolution. A Fourier transform takes a function from one domain, into another, with a two dimensional Fourier transform given by

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(ux+vy)}dxdy\,. \tag{23}$$

with the inverse of the Fourier transform

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(ux+vy)}dudv\,. \tag{24}$$

It is typical convention to denote the Fourier transform of a function through $\mathcal{F}\{\}$ and the inverse through $\mathcal{F}^{-1}\{\}$.

The convolution of two functions $f(x)$ and $g(x)$ is given by

$$(f \star g)(x) = \int_{-\infty}^{\infty} f(\tau)\cdot g(x-\tau)d\tau\,. \tag{25}$$

It is convention to denote the convolution of two functions through the use of $\star$ as to distinguish it from multiplication $(*)$. A convolution in one domain corresponds to multiplication in the other, which can be seen by taking the Fourier transform of a convolution

$$\mathcal{F}\{f \star g\}(\nu) = F(\nu)G(\nu)\,. \tag{26}$$

with $F = \mathcal{F}\{f\}$ and G $= \mathcal{F}\{g\}$. This information is sufficient to begin the discussion on interferometry.

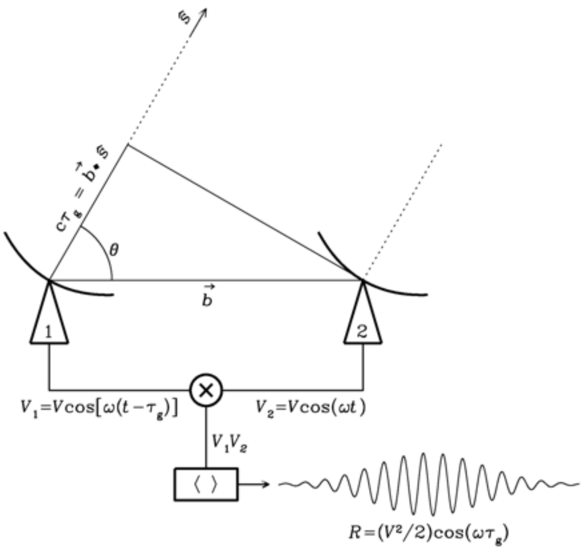
2.3.2 Interferometry: Response and Visibilities

The simplest way to describe how an interferometer works is by considering a two-element interferometer, observing a point source emitting monochromatic plane waves. Let $\hat{s}$ denote the direction of the point source in the sky and $\vec{b}$ the baseline of the telescopes. Figure 4 displays the current setup. To note, both $\vec{b}$ and $\hat{s}$ are three dimensional quantities, with $\vec{b} = (b_x, b_y, b_z)$ and $\hat{s} = (l, m, n)$[8].

If $\hat{s}$ is at an angle θ relative to the ground, the telescopes of the interferometer will have point at the same angle to view the source. As a result of this pointing angle, the distance the electromagnetic radiation must travel to each telescope differs. As a geometrical consequence, there will be a time delay between when each telescope receives the same electromagnetic radiation. This delay is referred to as the geometric delay and is denoted by τ_g [17]. Now, the extra path length travelled by the one ray is given by $\vec{b}\cdot\hat{s}$, where the geometric delay is related to this path length by [18, 17]

$$\tau_g = \vec{b}\cdot\hat{s}/c = b\sin\theta/c \tag{27}$$

where b is the distance between the telescopes. Since electromagnetic radiation oscillates, a voltage is induced on the antenna of each telescope. As a result of the geometric time delay, the same portion of electromagnetic radiation will



$\hat{s}$
$c\tau_g = \vec{b}\cdot\hat{s}$
θ
$\vec{b}$
1
2
$V_1 = V\cos[\omega(t-\tau_g)]$
$V_2 = V\cos(\omega t)$
V_1V_2
⟨ ⟩
$R = (V^2/2)\cos(\omega\tau_g)$

Figure 4: Simple Two-Element Interferometer (sourced from [18])

[8] l, m and n are direction cosines.

arrive at each antenna at differing times. This time difference will be present within the induced voltages as well, where [17]

$$V_1(t) = V\cos(\omega[t-\tau_g]) \ \text{ and } \ V_2(t) = V\cos(\omega t)\,, \tag{28}$$

with ω the angular frequency of the light, corresponding frequency $\nu = \omega/2\pi$ and wavelength $\lambda = c/\nu$.

These voltages are then sent to a device referred to as the correlator, where they are multiplied and time averaged. The result is referred to as the response of the correlator, and is mathematically defined as [18]

$$R = \langle V_1 V_2 \rangle = \left(\frac{V^2}{2}\right)\cos\left(\omega\tau_g\right) \tag{29}$$

where using the the expression for τ_g

$$R_{\mathrm{c}} = \left(\frac{V^2}{2}\right)\cos\left(2\pi\vec{b}\cdot\hat{s}/\lambda\right)\,. \tag{30}$$

This is the real response of a multiplying correlator, which may be denoted as R_{c}. There is also an imaginary component, however, this will only be introduced when looking at an extended source. For now, it is sufficient know that there exists both a since and cosine component.

These simplified assumptions allow for the above results to be generalized for sources that are no longer point sources, and where the electromagnetic radiation is incoherent (the radiation emitted from the source is random and uneven). Hence, it is essential then to introduce $I_\nu(\hat{s})$, the sky brightness distribution. This is the mathematical description of how bright each portion of the sky is, at a particular frequency ν. The sky brightness distribution is only concerned with the portion of sky to be imaged. Since $I_\nu(\hat{s})$ represents the physical sky, $I_\nu(\hat{s})$ has to be a real function. As a result, $I(\hat{s})$ may be broken up into an odd and an even contribution. If $I_{\mathrm{o}}(\hat{s})$ denotes the odd contribution, and $I_{\mathrm{e}}(\hat{s})$ denotes the even contribution, then $I(\hat{s})$ may be written as [18]

$$I(\hat{s}) = I_{\mathrm{o}}(\hat{s}) + I_{\mathrm{e}}(\hat{s})\,. \tag{31}$$

The response R_{c} of a simple interferometer to an incoherent, slightly extended source is obtained by treating the object as a sum of point sources [18]

$$R_{\mathrm{c}} = \int I_\nu\left(\hat{s}\right)\cos\left(2\pi\vec{b}\cdot\hat{s}/\lambda\right)d\Omega\,. \tag{32}$$

Since cosine is an even function, the odd contributions of $I_\nu(\hat{s})$ will cancel under the integral. Thus, a sine correlator is required to account for the odd contribution of $I(\hat{s})$. The corresponding response of a sine correlator in integral from is given by

$$R_{\mathrm{s}} = \int I\left(\hat{s}\right)\sin\left(2\pi\vec{b}\cdot\hat{s}/\lambda\right)d\Omega\,. \tag{33}$$

Now, with the use of equation (33) and (34), the complex visibility may be defined as [18]

$$V = R_{\mathrm{c}} - iR_{\mathrm{s}} = \int I(\hat{s})\left[\cos\left(2\pi\vec{b}\cdot\hat{s}/\lambda\right) - i\sin\left(2\pi\vec{b}\cdot\hat{s}/\lambda\right)\right]d\Omega \tag{34}$$

and simplifying with Euler's formula yields

$$V = \int I(\hat{s})\exp\left[-i2\pi\vec{b}\cdot\hat{s}/\lambda\right]d\Omega\,. \tag{35}$$

This is a very important equation in interferometry, as it is what the interferometer measures when pointing at a particular source in the sky.

The following is introduced, which becomes vital in plotting/analysing visibilities. The term $\vec{b}/\lambda$ is then set to $\vec{b}/\lambda = (u,v,w)$. Note that $u = b_x/\lambda$, $v = b_y/\lambda$ and $w = b_z/\lambda$, where u, v and w are dimensionless quantities. (u,v,w) form the basis of a three-dimensional coordinate system. Each parameter is set to a particular convention. w is set to point in the direction of $\hat{s}_0$. u points east within the plane tangential to $\hat{s}_0$, and v points north in that same plane [17].

Now, a infinitesimal solid angle is given by

$$d\Omega = \sin\theta d\theta d\phi \tag{36}$$

and using $l = \sin\theta\cos\phi$, $m = \sin\theta\sin\phi$ and $n = \cos\theta$ the infinitesimal solid angle may be written as

$$d\Omega = \frac{dldm}{\sqrt{1 - l^2 - m^2}} \tag{37}$$

(this result can also be found in [17]). The dot product $\vec{b}/\lambda \cdot \hat{s}$ is given by

$$\left(\vec{b}/\lambda\right) \cdot \hat{s} = (u, v, w) \cdot (l, m, n) = ul + vm + wn \tag{38}$$

With regards to how $\hat{s}$ is defined, the only relevant properties of $\hat{s}$ to $I_\nu(\hat{s})$ are l and m. n does not serve any relevance to I_ν as the sky brightness distribution is a two dimensional projection of the portion of sky being viewed. Hence, the sky brightness distribution is a two-dimensional function, representing the viewed portion of sky as a plane. $I_\nu(\hat{s}) = I_\nu(l, m)$ and the complex visibility is

$$V(u, v, w) = \int\int \frac{I_\nu(l, m)}{\sqrt{1 - l^2 - m^2}} \exp\left[-i2\pi(ul + vm + wn)\right] dldm\,. \tag{39}$$

If $w\theta^2 \ll 1$ is met, and the baselines are approximately coplanar, then it is sufficient to approximate w to be zero [18]. Thus, equation (40) then becomes

$$V(u, v) = \int\int \frac{I_\nu(l, m)}{\sqrt{1 - l^2 - m^2}} \exp\left[-i2\pi(ul + vm)\right] dldm\,. \tag{40}$$

The above equation is a two-dimensional Fourier transform of the brightness distribution. These visibilities are what interferometers measure. If there was a full uv plane of visibilities, then performing the inverse Fourier transform of the uv plane would result in the full sky brightness distribution.

$$\frac{I_\nu(l, m)}{\sqrt{1 - l^2 - m^2}} = \int\int V(u, v) \exp\left[i2\pi(ul + vm)\right] dudv\,. \tag{41}$$

2.3.3 Imaging and Image Processing

Due to physical limitations, the uv plane cannot be fully sampled. However, there are a variety of ways and techniques to do so. Some of these techniques include using multiple baselines, moving the baselines, using the Earth's rotation and observing at different frequencies. The more baselines present within an array, the more points on the uv plane are sampled. Similarly, using the Earth's rotation allows for tracks to be made within the uv plane. The other two methods depend on physically changing the baselines. Observing at different frequencies allows for the filling of the uv plane by changing the baselines through their wavelength dependency, while moving the actual telescopes physically puts the baselines at different locations on the uv plane. Employing these methods in isolation or as a combination results in a better filled uv plane.

The uv plane contains a discrete number of sampled visibilities. These discretely sampled points may be mathematically expressed as [19]

$$S(u, v) = \sum_i \delta(u - u_i, v - v_i) \tag{42}$$

where $\delta(u - u_i, v - v_i)$ is a Dirac delta function, and (u_i, v_i) are all the points at which a visibility was sampled. This sampling function multiplied with the visibility function, $V(u, v) * S(u, v)$ represents the collected data from the interferometer. The key point of interest is acquiring the sky brightness distribution, and this is done by taking the inverse Fourier transform of $V(u, v) * S(u, v)$, where

$$\mathcal{F}^{-1}\{V(u, v) \cdot S(u, v)\} = I_\nu(l, m) \star B(l, m) = I_\nu^D(l, m) \tag{43}$$

with

$$I_\nu(l, m) = \mathcal{F}^{-1}\{V(u, v)\} \text{ and } B(l, m) = \mathcal{F}^{-1}\{S(u, v)\}. \tag{44}$$

Here, $I_\nu^D(l,m)$ is referred to as the dirty image, while $B(l,m)$ is referred to as the point spread function[9] [20]. It is clear from the above that the dirty image is the convolution of the sky brightness distribution and the point spread function. The key idea is to separate the point spread function from sky brightness distribution. This idea is executed through the CLEAN algorithm.

[21] introduces an equation of the form

$$I_D(l,m) = \sum I_i \, B(l - l_i, m - m_i) + I_\epsilon(l,m), \tag{45}$$

where $I_\varepsilon(l,m)$ is referred to as the residual image, and the I_i elements represent point sources. The job of the CLEAN algorithm, as stated by [21], is to approximate the sky brightness distribution through a superposition of finite point sources, located at (l_i, m_i), such that I_i satisfies equation (46). What this equation means in the context of CLEAN is that the dirty image is the sum of all the point sources I_i at the positions (l_i, m_i), added to the residual image.

[22] introduces the CLEAN algorithm as an iterative procedure that builds a model image of the sky brightness distribution, using both the dirty image and the point spread function to do so. Although the algorithm mentioned in [22] is somewhat simple in comparison to the CLEAN algorithms of today, it forms the basis of image processing. A more modern version of CLEAN is described by [23], and is summarized as follows: CLEAN consists of two main cycles termed the minor and major cycles. The minor cycle is similar to the description of CLEAN as mentioned by [22], taking place only in the image plane. The major cycle is more complicated, taking place in both the *uv* plane and the image plane.

In the minor cycle, the brightest point on the residual[10] is located and removed, being placed at an identical position on the model images with a scaled intensity. A copy of the model image is convolved with the point spread function and subtracted from the residual. This is then repeated a number of times. However, the residuals produced through the minor cycle are not as accurate as the residual produced by the major cycle.

In the major cycle, the model image is taken from the image plane, to the *uv* plane. This is done by Fourier transforming the model image to produce model visibilities. The model visibilities are then subtracted from the original *uv* data, which results in a new residual. An inverse Fourier transform is then performed on the residual visibilities, to produce a new residual in the image plane. The major cycle is more accurate than the minor cycle, but is more computationally taxing. Hence, the minor cycle is run for a number of iterations before an iteration of the major cycle is performed.

Once a sufficient number of cycles has been completed, the clean image is obtained by convolving the model image with the clean beam. The clean beam, often referred to as the restoring beam, is simply a Gaussian fitted to the main lobe of the point spread function. One thing to note however, is that the image produced is dependent on the number of iterations chosen. Too many iterations, and noise and artifacts may be added into the image. Too few iterations, and the image may not accurately reflect the more fine scale structure of the source, with additional side-lobes present.

2.3.4 Interferometry: Resolution and Sensitivity

Angular resolution may be defined as the minimum angular separation required to distinguish details of an object, or even the ability to resolve multiple separate objects of interest. Sensitivity describes the faintest brightness an interferometer can detect. Higher sensitivity allows for fainter surface brightnesses to be observed, while lower sensitivity correlates to fainter components being overshadowed by noise. The angular resolution and the sensitivity of an interferometer is formulated by [18] as follows:

The best angular resolution an interferometer can achieve is given by

$$\theta \approx \frac{\lambda}{D_{\max}} \tag{46}$$

where λ is the wavelength of light, and $D_{\max}$ is the largest baseline within the array of N dishes.

[9] The psf (point spread function) may also be referred to as the dirty beam.

[10] In the first iterative step, the residual is simply the dirty image. After a component has been removed, what remains is then called the residual. In the next iteration, a component is removed from the residual, and what remains is a new residual.

Following this, the RMS noise for interferometer is given by

$$\sigma_S = \frac{2kT_S}{A_e \left[N(N-1)\Delta\nu\tau\right]^{1/2}} \tag{47}$$

with T_S the system temperature, k the Boltzmann constant, A_e the effective collecting area of the dish, N the number of telescopes, $\Delta\nu$ the bandwidth and τ being the integration time. The rms noise value sets the limit for sensitivity of the interferometer. Any surface brightness value from the sky less than σ_S will be hidden underneath the rms noise.

2.4 The SKA

In the current phase of construction, the SKA consists of two telescopes referred to as SKA-Low and SKA-Mid. SKA-Mid consists of 197 dishes [15]. The total number of dishes is a contribution of 64 13.5 meter dishes from the already existing MeerKAT, and 133 15 meter dishes that will be constructed around MeerKAT. SKA-Mid will operate in the frequency range of 0.35 gigahertz to 15.3 gigahertz. The band of specific interest is band 5a, which ranges from 4.6 gigahertz to 8.5 gigahertz, with a central frequency of 6.55 gigahertz [15]. SKA-Mid also proposes a larger maximum baseline than that of MeerKAT alone, with the longest baseline measuring 150 kilometers [15].

3 Simulations

3.1 The Light Curves

The foundation of the project was built upon simulating a master light curve[11]. The curve was generated using a superposition of four sine waves, each waving having a distinct amplitude, frequency and phase. The curve was modelled in a way such what it would bare resemblance to the observed light curve of the gravitational lens system B1608+656 [10]. The master light curve was normalised to have an average flux of one over a period of 442.9 days[12].

To model the light curves of the four objects, parameters such as the average flux, position and time delays were required. Instead of choosing random values, the parameters were based off of data found within [24]. To note, the values used were slightly modified, such as the average flux, the time delays and the positions of the object. The parameters used in producing the object light curves are found in Table 1.

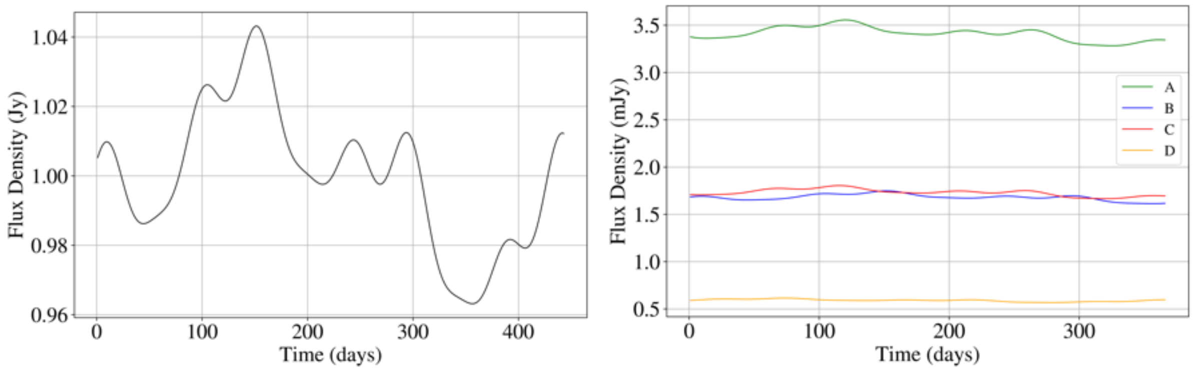
Combined Results			
Object	Average Flux (mJy)	Delay (day)	Position (J2000)
A	3.41	$\tau_{AB} = 31.5$	00°00’00.0” -45°00’00.0”
B	1.68	$\tau_{BB} = 0$	-00°00’00.738” -45°00’01.962”
C	1.73	$\tau_{CB} = 36.2$	-00°00’00.745” -45°00’00.453”
D	0.590	$\tau_{DB} = 77.9$	00°00’01.130” -45°00’01.257”

Table 1: Light Curve Properties.

Figure 5 below displays the master light curve (a), and the object light curves (b). As per Figure 5(a), the master light curve was generated over 442.9 days. The maximum time delay was 77.9 days and added to 365 days is 442.9 days. This ensured that each curve could be simulated over a year. Each curve was generated in the following way: The master light curve was sampled on a specific interval based on the image time delay. For a time delay τ_d, the interval chosen for sampling was then $[\tau_\text{d}, \tau_\text{d} + 365]$. Each interval, irrespective of time delay, was 365 days long. Each curve was then shifted such that it began on day 0. This is reflected in Figure 5(b), which shows that the light curves were embedded with their respective delays while possessing the same characteristic shapes, with amplitudes determined by their respective average fluxes.

[11] All of the other light curves were base off of this simulated light curve and is aptly termed the master light curve.

[12] The actual function is defined on the interval $[0, 2\pi]$.



(a) Master Light Curve

(b) Object Light Curves

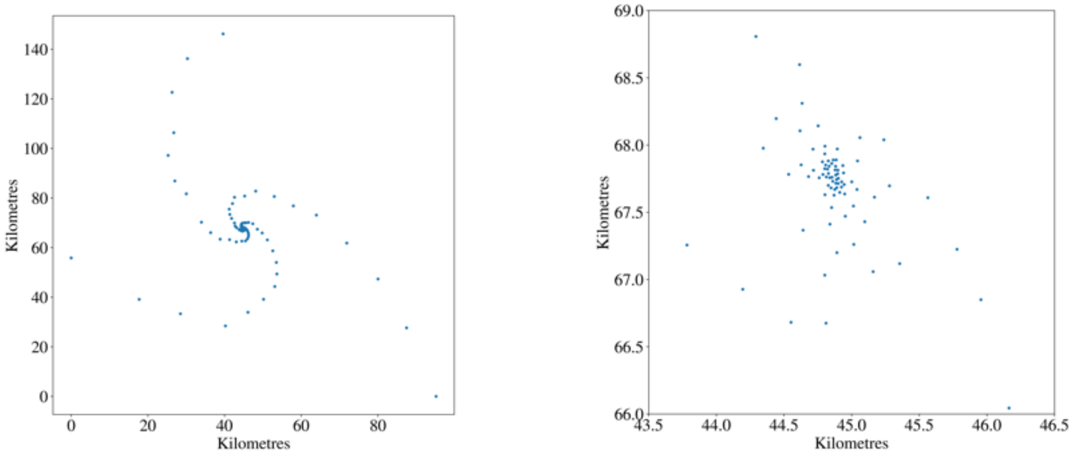
Figure 5: Generated Curves.

3.2 CASA

With the data for the object light curves prepared, simulations of interferometric observations of these objects were performed. This was done through the use of the Common Astronomy Software Application package (The Casa Team et al., 2022) [25], simply referred to as CASA. CASA provided a variety of tools and tasks which allowed for the interferometric observation simulations of the images, whose documentation is available at [26, 27].

The lensed images were simulated through the use of a componentlist. In this instance, the sky brightness was modelled using four point sources. These four point sources were added to one componentlist file as points sources, with thier respective positions and fluxes as additional input. A flux reading for each object was taken from it's respective curve for each day of the year, when the objects were in the transit position[13]. This required an individual componentlist for each day in the year. Thus, one componentlist file, a .cl file, contained four objects and their positions, with one flux value for each object.

Interferometric simulations were then performed on each of these .cl files through the use of a casatask called simobserve. Simobserve simulates the visibilities of the given array, creating measurement sets, stored in .ms files. There were a few essential arguments that simobserve required. It required the componentlist of interest, the antenna list, the hour angle and the total time dedicated to the observation. The hour angle argument was set to transit, the total time was set to 45 seconds and the antenna list argument was passed the array configuration. The array used was the SKA AA4 dish configuration. The array configuration is found in Figure 6.



(a) Full AA4 Configuration

(b) Inner Region AA4 Configuration

Figure 6: SKA AA4 15m Dish Configuration.

[13] The transit specification was set in the simobserve casatask

It is important to note that the configuration file used was not a part of the standard configurations found in CASA. Thus, CASA does not have the sensitivity calibrations for the configuration at hand. The sensitivity thus had to be added to the produced .ms file. This was done using the SKAO sensitivity calculator [28] and the casatool setnoise. Using the calculator, the weighted continuum sensitivity was found to be 4.96 µJy/beam for a robust value of 2. This would then suggest that when looking at a blank portion sky, the RMS noise level of the image should then also be 4.96 µJy/beam. Setnoise then needed an argument called simplenoise that corresponded to this RMS noise of the image. This was done by passing in a component list containing one object with zero flux to simobserve. Simobserve then produced a .ms file corresponding to a blank observation. The casatask tclean[14] was then used to generate an image, where the cube analysis and rendering tool for astronomy (CARTA) [29] was used to measure the RMS noise of the image. Simple noise was adjusted accordingly until the RMS of 4.96 µJy/beam was reached. The corresponding simple noise value σ_{sn} was found to be $\sigma_{\text{sn}} = 0.5558$ mJy[15].

Once the appropriate noise was added, the .ms file was then passed to tclean. tclean constructs an image from the given visibilities and from this image, a model of the sky may be reconstructed. The tclean task does this through the CLEAN algorithm as described in Section 3. tclean required a number of arguments, the most relevant of these being the visibilities, the deconvolver method, the weighting, the corresponding robust value, a mask, the threshold and the number of iterations. The visibilities are supplied through the .ms file. The deconvolver argument assigns a specific minor cycle method to the CLEAN algorithm, where the method 'hogbom' was passed. This is the same method as described in [22]. Weighting is the weighting scheme for the visibilities, where the 'Briggs' scheme was chosen. The robust argument is associated with the 'Briggs' weighting scheme, and robust -0.5 was used. The mask argument accepts a mask[16] for which the cleaning will be done in[17]. This was chosen based on the positions of the objects. Threshold acts as a termination condition to the CLEAN algorithm. tclean will clean within the mask until the desired threshold value is reached. The chosen value for the threshold was $3\sigma_{\text{RMS}}$ to ensure that tclean wasn't mistaking possible noise as additional sources. Finally, niter is another termination condition based on a maximum number of iterations the CLEAN algorithm operates for. In this instance, it was set to a high enough value such that the threshold was responsible for terminating the cleaning process.

tclean outputs a number of files, with the extensions .image, .model, .pb, .psf, .residual and .sumwt. For each day of the year, this collection of files is produced. The .image file is the reconstructed sky model. The .model file is the model image all of the clean components are added to. The .psf is the point spread function, or the dirty beam. The last file of significant note is the residual, which is what remains of the dirty image after cleaning. CARTA [29] was used to visualise these files. Figure 7 displays the contents of each file. These were generated by converting each file type to a fits through CASA, and converting the fits file to a .png using astropy [30, 31].

Imfit was then used to determine the flux and flux error associated with each object. imfit is a casatask which fit's a Gaussian to specified regions in the image. This then allows the total flux, along with the error associated with the flux measure to be determined. What imfit returns is a python dictionary. For each observation day, the flux and it's associated error were collected for each object. This was the data required to carry out the time delay analysis.

3.3 PyCS3 and the Time Delay Estimations

The time delay analysis was done using the PyCS3 package (Tewes et al. 2013; Millon et al. 2020b) [32, 33], from which the time delays and associated error was produced. All the information on the functionality of PyCS3, along with the subpackages and functions involved was obtained from the official documentation [34].

PyCS3 required that the flux data be converted to magnitudes. The flux was converted using the AB magnitude system defined as [35] and given by

$$m(f_\nu) = -2.5\log_{10}\left(\frac{f_\nu}{3631\,\text{Jy}}\right) = -2.5\log_{10}(f_\nu) + 8.90\,. \tag{48}$$

[14] More will be discussed on tclean further on.

[15] This simple noise value was then used for all the other robust values.

[16] A mask is a region/regions for which the CLEAN algorithm is forced to work in. As a result, no cleaning is done outside the area of the mask.

[17] This prevents CLEAN from creating CLEAN components where there aren't any sources. The reason it may do so could be due to higher than expected noise in a certain region.

with f_ν denoting the flux of the object. The error associated with each flux value also had to be converted into magnitudes. This was done by using propagation of error [36],

$$\sigma_m = \left|\frac{\partial m}{\partial f_\nu}\right| \sigma_{f_\nu} = \left|\frac{2.5}{f_\nu \ln(10)}\right| \sigma_{f_\nu} \tag{49}$$

where σ_{f_ν} is the flux error and σ_m is the error in magnitudes.

The light curve data was imported through the lc_func, using the function rdbimport. Here, rdb refers to a particular way in which the files containing the magnitude are formatted. The function shifttime was then used to align the curves to the correct time delay. As this was a simulation, the actual time delays of each curve were known. In reality however, the correct time delays are generally guessed based on the collected data. All the curves were shifted relative to object B. Hence, the shifttime arguments were 31.5, 0, 36.2 and 77.9 corresponding to the objects A, B, C and D respectively.

PyCS3 offered two optimisers, that being the spline optimiser and the regdiff optimiser. The spline optimiser fits a piecewise polynomial, referred to a spline in this instance, to the data. The degree to which this is done is controlled by objects referred to as knots, which are either fixed or free [37]. The regdiff optimiser stands for regression difference. The method works through aligning the curves in such a manner that finds the minimum variability in the difference of their magnitudes. The regression use is Gaussian process regression.

After having passed the light curves and shifting them, a spline was fit to the data using the functions opt_rough and opt_fine from the spl.topopt package. The first function opt_rough generates a rough spline, where opt_fine refines the spline. Figure 4 below shows the fitted spline[18]. The generated spline would then act as the intrinsic curve when generating the mock curves. A mock in this instance is a curve with the basic shape of the spline[19], with the characteristics of the object light curves. These characteristics include things such as the time shifts, the error bars, observation cadence, and so on.

The function used to generate all of these mock curves was multidraw, under the module sim.draw. There were a variety of parameters that multidraw required. The light curves and the associated spline were passed as arguments. The number of generated mocks had to be specified, with 50 being passed[20]. The argument shotnoise selected a method for adding random variations to the simulated curve. The method chosen was 'mcres' which stands for monte carlo residual.

Once the mocks had been created, they were then passed to the function multirun, under the module sim.run. Multirun then determined all the time delays for the mock curves. It produced relative time delays and the error associated with each one of these delay values. Multirun required a few parameters. The most important of these was the mock curves. Another important parameter was the type of optimisation method to use. Both spline and regdiff had to be used, however, multirun could only use one at a time. Thus, multirun was run twice, once with the spline optimisation, and once with the regdiff optimisation.

4 Results

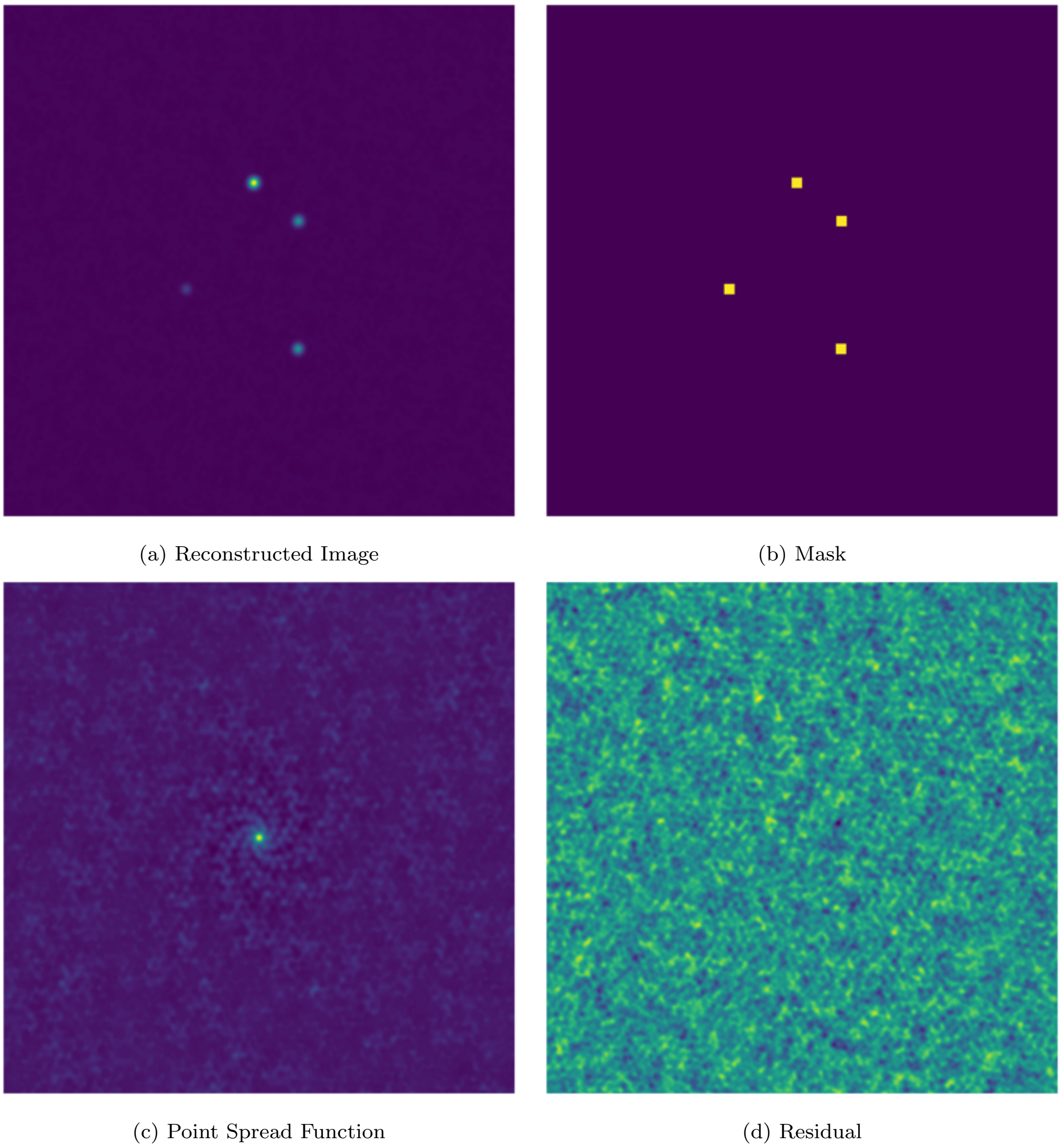
Figure 7 displays the images produced by tclean. Figure 7(b) is the mask used during cleaning. Looking at Figure 7(d), it is clear that the cleaning was successful, having removed all of the source components. Figure 7(a) displays the reconstructed image, showing four distinct objects. In other words, the lensed images are well resolved, with a restoring beam of size $0.10569'' \times 0.10144''$.

Figure 8 displays the model light curve and the fitted spline lines to the object light curves with respect to the cadence of observation. The spline was fitted by adjusting each individual object light curve in both time and magnitude, such that the characteristic shape of each aligned. The vertical bars within each plot represent the error associated with each flux value. The largest error bars present are from D, the faintest of the lensed image, whilst the smallest error bars are present are from A, the brightest of the objects. The error bars of object B and C are similar, as both have a similar brightness. This can be verified by looking at the Figure 7(a).

[18] To note, the spline shown if Figure 4 was the spline for the one observation every day of the year.

[19] The spline is what provides the magnitudes to the mock curves

[20] Each mock corresponded to a simulation of the 4 light curves passed into the function.



(a) Reconstructed Image

(b) Mask

(c) Point Spread Function

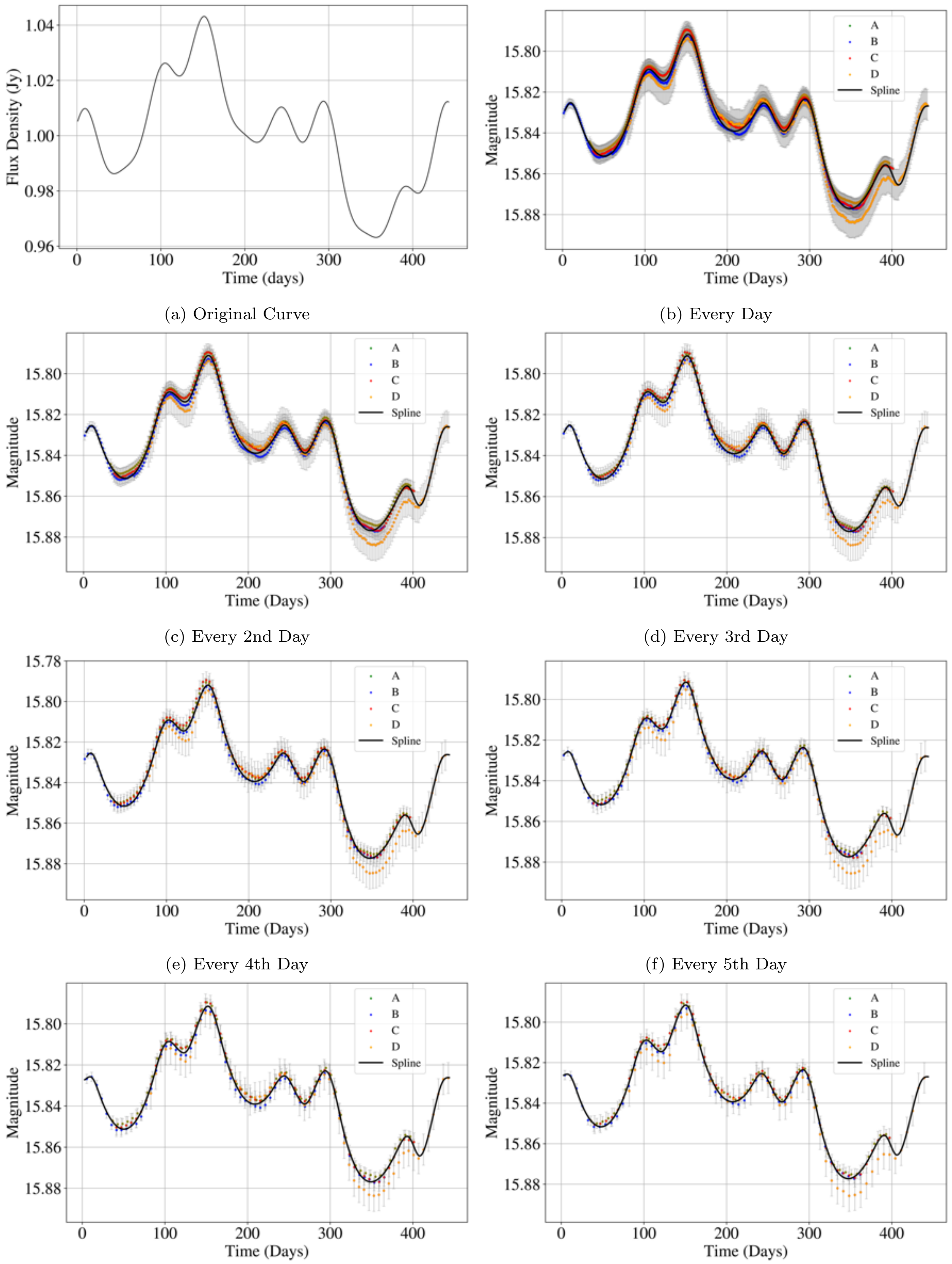
(d) Residual

Figure 7: Images Produced by tclean

One of the first things of notice is that each of the spline curves generated for a regular cadence look near identical with regards to the shape and magnitude. With comparison to the original light curve, each spline curve has rounder troughs. This rounding is most evident around 50 days, 200 days and 350 days. In addition to this, the ends of each spline curve flatten out as the cadence of the observations decreases, with the full year of observations (Figure 8(b)) most closely resembling the model light curve. With that said, all of the produced splines closely resemble the model light curve, with lessening precession as cadence decreases. There is one portion of each spline however that is not scaled correctly with regards to the model curve.

The right end of each spline is constructed purely based off of the data from the light curve of object D. From each image in Figure 8, it is clear that object D's data is the least similar in term of scale to the rest of the light curves, with the dissimilarities beginning from 310 days and continuing onwards. As D shares the same characteristic shape as the rest of the curves, the portion of spline based purely on D should accurately reflect the shape of the intrinsic variability of the object, however, at not quite the correct scale. This happens to be the case when looking at Figure 8(a). The trough in each spline around day 410 is too deep. In comparison to the model light curve, the correct scaling seems to lies within the upper bounds of object D's errors.

Figure 9 displays the spline fittings of irregularly sampled data along with a corresponding histogram.



(a) Original Curve

(b) Every Day

(c) Every 2nd Day

(d) Every 3rd Day

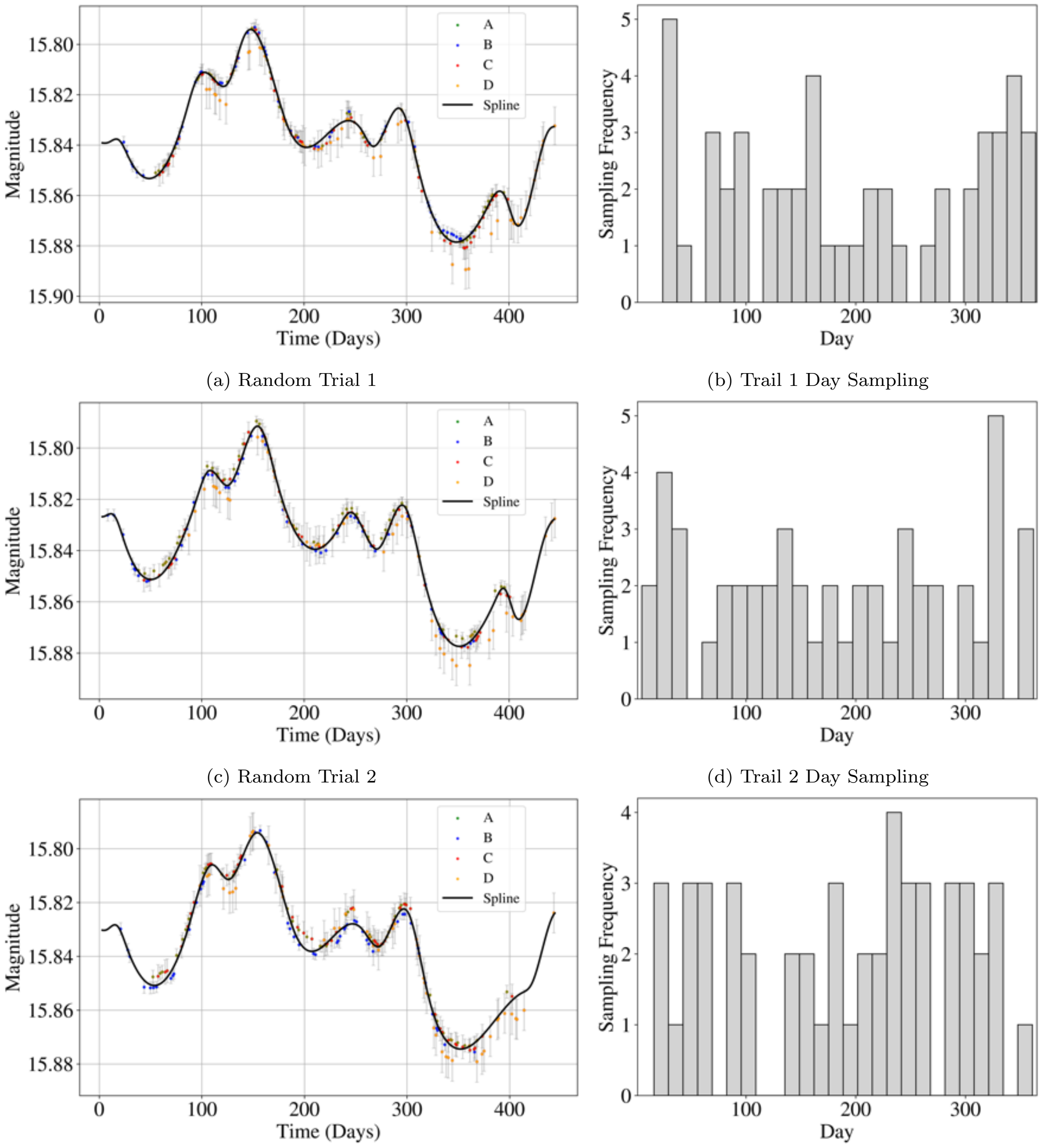
(e) Every 4th Day

(f) Every 5th Day

(g) Every 6th Day

(h) Every 7th Day

Figure 8: Fitted Spline Per Regular Observation Cadence



(a) Random Trial 1

(b) Trail 1 Day Sampling

(c) Random Trial 2

(d) Trail 2 Day Sampling

(e) Random Trial 3

(f) Trial 3 Day Sampling

Figure 9: Random 50 Days of Observations

Cadence	Daily	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}
Spline Results							
Δ BA	$31.44^{+0.27}_{-0.27}$	$31.41^{+0.33}_{-0.33}$	$31.42^{+0.34}_{-0.34}$	$31.52^{+0.34}_{-0.34}$	$31.49^{+0.37}_{-0.37}$	$31.45^{+0.46}_{-0.46}$	$31.51^{+0.38}_{-0.38}$
Δ BC	$35.84^{+0.25}_{-0.25}$	$35.82^{+0.31}_{-0.31}$	$35.83^{+0.33}_{-0.33}$	$35.87^{+0.38}_{-0.38}$	$35.86^{+0.34}_{-0.34}$	$35.81^{+0.43}_{-0.43}$	$35.87^{+0.46}_{-0.46}$
Δ BD	$77.63^{+0.36}_{-0.36}$	$77.46^{+0.43}_{-0.43}$	$77.54^{+0.47}_{-0.47}$	$77.59^{+0.63}_{-0.63}$	$77.81^{+0.78}_{-0.78}$	$77.52^{+0.75}_{-0.75}$	$77.74^{+0.86}_{-0.86}$
Δ AC	$4.4^{+0.13}_{-0.13}$	$4.4^{+0.22}_{-0.22}$	$4.4^{+0.23}_{-0.23}$	$4.36^{+0.29}_{-0.29}$	$4.37^{+0.28}_{-0.28}$	$4.37^{+0.34}_{-0.34}$	$4.36^{+0.39}_{-0.39}$
Δ AD	$46.19^{+0.34}_{-0.34}$	$46.05^{+0.45}_{-0.45}$	$46.11^{+0.48}_{-0.48}$	$46.08^{+0.59}_{-0.59}$	$46.31^{+0.8}_{-0.8}$	$46.07^{+0.75}_{-0.75}$	$46.23^{+0.85}_{-0.85}$
Δ CD	$41.79^{+0.33}_{-0.33}$	$41.64^{+0.43}_{-0.43}$	$41.71^{+0.46}_{-0.46}$	$41.72^{+0.6}_{-0.6}$	$41.94^{+0.78}_{-0.78}$	$41.71^{+0.73}_{-0.73}$	$41.87^{+0.85}_{-0.85}$
Regdiff Results							
Δ BA	$31.43^{+0.19}_{-0.19}$	$31.48^{+0.3}_{-0.3}$	$31.4^{+0.34}_{-0.34}$	$31.51^{+0.32}_{-0.32}$	$31.42^{+0.39}_{-0.39}$	$31.55^{+0.56}_{-0.56}$	$31.55^{+0.51}_{-0.51}$
Δ BC	$35.88^{+0.19}_{-0.19}$	$35.97^{+0.34}_{-0.34}$	$35.88^{+0.3}_{-0.3}$	$35.97^{+0.45}_{-0.45}$	$35.93^{+0.33}_{-0.33}$	$36.07^{+0.52}_{-0.52}$	$35.95^{+0.59}_{-0.59}$
Δ BD	$77.49^{+0.33}_{-0.33}$	$77.56^{+0.41}_{-0.41}$	$77.46^{+0.47}_{-0.47}$	$77.63^{+0.56}_{-0.56}$	$77.59^{+0.83}_{-0.83}$	$77.48^{+0.73}_{-0.73}$	$77.45^{+0.88}_{-0.88}$
Δ AC	$4.44^{+0.2}_{-0.2}$	$4.49^{+0.33}_{-0.33}$	$4.48^{+0.32}_{-0.32}$	$4.46^{+0.45}_{-0.45}$	$4.5^{+0.4}_{-0.4}$	$4.52^{+0.58}_{-0.58}$	$4.4^{+0.6}_{-0.6}$
Δ AD	$46.06^{+0.35}_{-0.35}$	$46.07^{+0.41}_{-0.41}$	$46.06^{+0.5}_{-0.5}$	$46.12^{+0.59}_{-0.59}$	$46.17^{+0.87}_{-0.87}$	$45.93^{+0.72}_{-0.72}$	$45.9^{+0.89}_{-0.89}$
Δ CD	$41.62^{+0.34}_{-0.34}$	$41.58^{+0.42}_{-0.42}$	$41.58^{+0.47}_{-0.47}$	$41.66^{+0.66}_{-0.66}$	$41.66^{+0.81}_{-0.81}$	$41.41^{+0.67}_{-0.67}$	$41.5^{+0.97}_{-0.97}$
Combined Result							
Δ BA	$31.46^{+0.24}_{-0.24}$	$31.46^{+0.32}_{-0.32}$	$31.42^{+0.32}_{-0.36}$	$31.54^{+0.36}_{-0.32}$	$31.46^{+0.36}_{-0.4}$	$31.5^{+0.48}_{-0.52}$	$31.54^{+0.44}_{-0.44}$
Δ BC	$35.82^{+0.24}_{-0.2}$	$35.86^{+0.32}_{-0.32}$	$35.82^{+0.32}_{-0.32}$	$35.9^{+0.4}_{-0.44}$	$35.86^{+0.36}_{-0.32}$	$35.9^{+0.48}_{-0.48}$	$35.9^{+0.52}_{-0.52}$
Δ BD	$77.55^{+0.32}_{-0.36}$	$77.47^{+0.44}_{-0.4}$	$77.47^{+0.48}_{-0.48}$	$77.59^{+0.6}_{-0.6}$	$77.67^{+0.8}_{-0.8}$	$77.47^{+0.76}_{-0.72}$	$77.59^{+0.84}_{-0.88}$
Δ AC	$4.39^{+0.2}_{-0.16}$	$4.43^{+0.28}_{-0.28}$	$4.43^{+0.28}_{-0.28}$	$4.39^{+0.36}_{-0.36}$	$4.43^{+0.32}_{-0.36}$	$4.43^{+0.44}_{-0.44}$	$4.35^{+0.52}_{-0.48}$
Δ AD	$46.1^{+0.36}_{-0.36}$	$46.02^{+0.44}_{-0.4}$	$46.06^{+0.48}_{-0.48}$	$46.06^{+0.6}_{-0.56}$	$46.22^{+0.84}_{-0.84}$	$45.98^{+0.72}_{-0.76}$	$46.06^{+0.84}_{-0.92}$
Δ CD	$41.66^{+0.36}_{-0.32}$	$41.58^{+0.44}_{-0.4}$	$41.62^{+0.48}_{-0.48}$	$41.66^{+0.64}_{-0.64}$	$41.78^{+0.8}_{-0.8}$	$41.54^{+0.68}_{-0.72}$	$41.66^{+0.92}_{-0.92}$

Table 2: Observation results by observation cadence (Briggs -0.5)

The histogram accompanying each spline fit has a total of 26 bins, each the size of roughly 14 days. This then groups the number of observations that take place within a two week span into one bin. Hence, the observation frequency[21] is based on a biweekly basis.

Figure 9(e) is the worst spline fit, with key characteristics of the model light curve missing or severely altered. This includes both flattened and extended peaks, with a trough at the end of the curve missing, replaced by what may be described as a slight kink. The histogram in Figure 9(f) shows an inconsistent observation frequency, with weeks and months without observations. Figure 9(a) shows a spline fit that more closely resembles the model light curve than what Figure 9(e) did. However, there are still key characteristics that are incorrectly scaled with regards to the model light curve, such as flattened peaks, and shallower troughs. Figure 9(b) shows the associated histogram. The key difference is that there aren't any monthly gaps, and that a large number of observations were performed towards the end of the year. Figure 9(c) is the most similar to the model light curve. The associated histogram, figure 9(d), shows somewhat of a more evenly distributed observation frequency. Combining this with the results of Figure 8, using a regular observation cadence leads to a well constructed spline, that more accurately mimics the model light curve.

Table 4 briefly summarises the input time delays. This serves as a reference of comparison for the results of Table 2 and 3.

[21] The observation frequency may also be referred to sampling frequency

Results	Run 1	Run 2	Run 3
Spline Results			
Δ BA	$31.68^{+0.38}_{-0.38}$	$31.13^{+0.56}_{-0.56}$	$31.52^{+0.78}_{-0.78}$
Δ BC	$35.69^{+0.43}_{-0.43}$	$35.36^{+0.46}_{-0.46}$	$36.38^{+1.03}_{-1.03}$
Δ BD	$81.24^{+2.17}_{-2.17}$	$79.22^{+1.07}_{-1.07}$	$77.41^{+1.09}_{-1.09}$
Δ AC	$4.01^{+0.41}_{-0.41}$	$4.23^{+0.39}_{-0.39}$	$4.86^{+0.89}_{-0.89}$
Δ AD	$49.56^{+2.22}_{-2.22}$	$48.09^{+1.12}_{-1.12}$	$45.89^{+1.38}_{-1.38}$
Δ CD	$45.55^{+2.23}_{-2.23}$	$43.86^{+1.11}_{-1.11}$	$41.03^{+1.44}_{-1.44}$
Regdiff Results			
Δ BA	$31.18^{+0.6}_{-0.6}$	$31.86^{+0.75}_{-0.75}$	$31.56^{+1.33}_{-1.33}$
Δ BC	$35.89^{+0.64}_{-0.64}$	$36.23^{+0.58}_{-0.58}$	$35.52^{+1.47}_{-1.47}$
Δ BD	$77.46^{+2.12}_{-2.12}$	$78.1^{+1.17}_{-1.17}$	$77.09^{+1.51}_{-1.51}$
Δ AC	$4.71^{+0.53}_{-0.53}$	$4.37^{+0.55}_{-0.55}$	$3.95^{+1.22}_{-1.22}$
Δ AD	$46.28^{+2.17}_{-2.17}$	$46.24^{+1.21}_{-1.21}$	$45.53^{+1.49}_{-1.49}$
Δ CD	$41.57^{+2.16}_{-2.16}$	$41.87^{+1.1}_{-1.1}$	$41.57^{+1.7}_{-1.7}$
Combined Results			
Δ BA	$31.5^{+0.6}_{-0.48}$	$31.46^{+0.68}_{-0.8}$	$31.54^{+1.0}_{-1.04}$
Δ BC	$35.74^{+0.56}_{-0.52}$	$35.7^{+0.8}_{-0.6}$	$35.98^{+1.24}_{-1.36}$
Δ BD	$79.31^{+3.0}_{-2.96}$	$78.67^{+1.2}_{-1.28}$	$77.27^{+1.24}_{-1.32}$
Δ AC	$4.31^{+0.64}_{-0.56}$	$4.27^{+0.48}_{-0.44}$	$4.47^{+1.08}_{-1.24}$
Δ AD	$47.86^{+2.86}_{-2.76}$	$47.18^{+1.48}_{-1.6}$	$45.7^{+1.4}_{-1.44}$
Δ CD	$43.5^{+3.12}_{-3.04}$	$42.82^{+1.56}_{-1.52}$	$41.26^{+1.6}_{-1.56}$

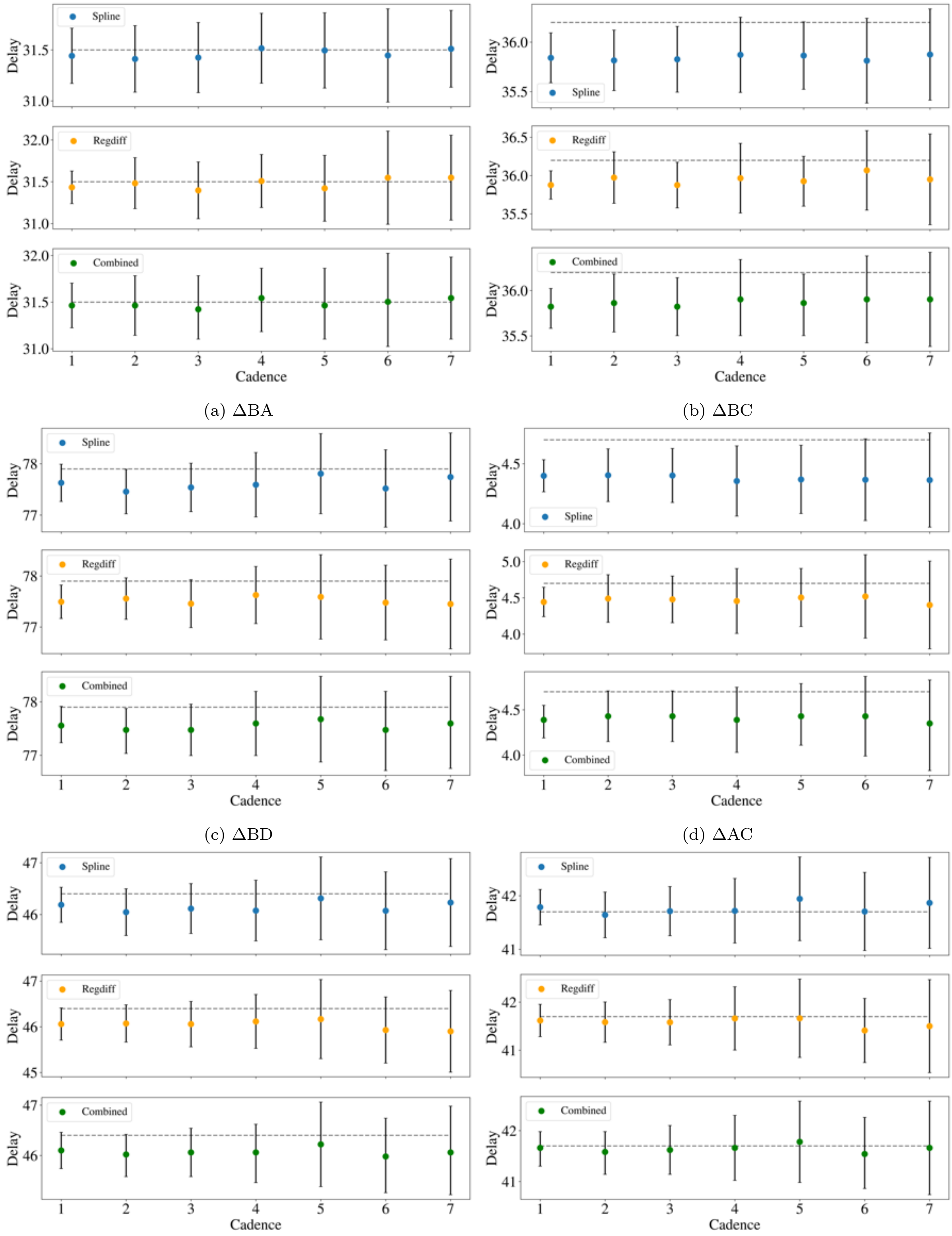
Table 3: Observation results by observation cadence (Briggs -0.5)

Relevance	Δ BA	Δ BC	Δ BD	Δ AC	Δ AD	Δ CD
Delay	31.5	36.2	77.9	4.7	46.4	41.7

Table 4: Input Delays

Table 2 displays the data obtained from PyCS3 for the even cadence observations. Each column represents observation cadence, where each row represents the relative time delay between the respective objects. With each value is an associated error range. The table consists of the results from the spline optimisation, the regdiff optimisation, and the combined results. Table 3 displays the observation data of the irregularly cadenced observations. The only structural difference between Table 2 and 3 is that each column represents the results of a random 50 days of observations. The data of Table 2 and 3 has been plotted in Figures 10 and 11 respectively.

Figures 10 and 11 share the same basic structure. All three results are present within each sub-figure, with the solid dots representing the estimated time delay value and associated error, and the dashed line representing the input time delay. The line serves as a comparison between the acquired result and the true result. The only structural difference between Figure 10 and 11 is in the x-axis. In figure 10, the x-axis represents the observation cadence, while in figure 11, the x-axis represents the random 50 day run. Having plotted the data from each table has allowed from a qualitative interpretation of the results.



(a) ΔBA

(b) ΔBC

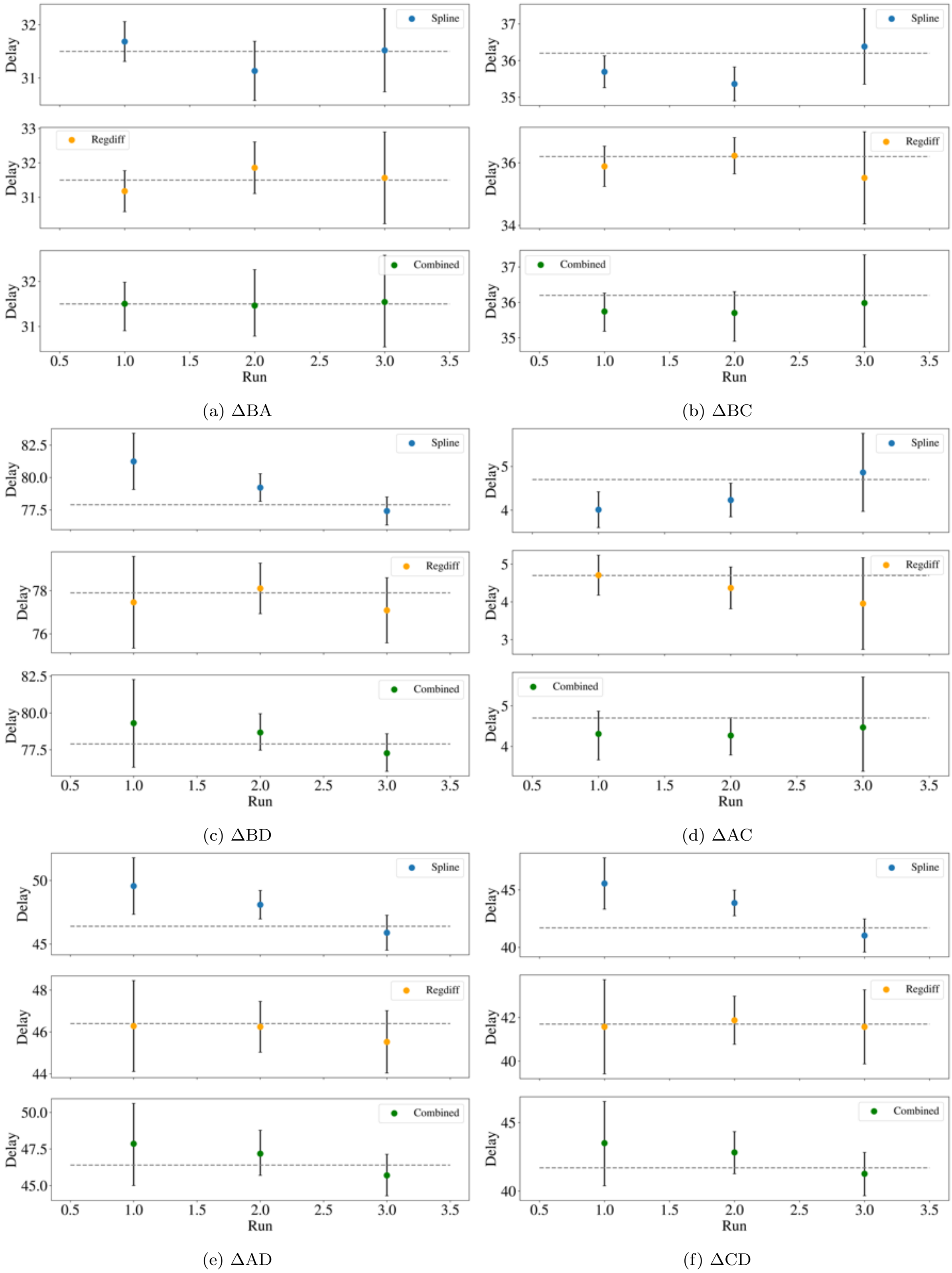
(c) ΔBD

(d) ΔAC

(e) ΔAD

(f) ΔCD

Figure 10: Visualisation of the from Table 2



(a) ΔBA

(b) ΔBC

(c) ΔBD

(d) ΔAC

(e) ΔAD

(f) ΔCD

Figure 11: Visualisation of the from Table 3

Refer to Figure 10 and Table 2. The first point of note is the general trend of the data, where with a decreasing cadence in the number of observations, there is an increase in the associated uncertainty of the estimated time delay. As a consequence, the least frequent cadence of observation had the greatest uncertainty in the estimated value of the time delay. There were outliers to this trend however. Some estimated delays at a higher cadence had a larger associated uncertainty than the succeeding delays estimations at a less frequent cadence. This includes ΔBA between cadence 6th and 7th, ΔBC between cadence 5th and 6th, and ΔAD between cadence 5th and 6th just to name a few. If such an outlier was present within the spline data, it was also present within the regdiff data, and subsequently, in the combined data results too.

The second point of notice is that a large majority of the estimated delays have uncertainty ranges which include the input delay. What is of particular intrigue however is that over half of the estimated time delays for the most highly cadenced observations have error ranges which do not contain the input delay. This is clear by referring to Figure 10. With regards to the spline data, the estimated time delays furthest from the input delays were ΔBC and ΔAC, where both the estimated values and the upper bound of the uncertainty lay below the input values. The corresponding regdiff data was closer to the actual delays, with the comparison line falling inside most of the upper error bounds.

Refer to Figure 11 and Table 3. Each optimiser showed varied performance over the different runs. The regdiff optimiser for runs 1 and 2 produced results that were closer to the input values than that of the spline optimiser. The uncertainties associated with each method were of a similar magnitude however. Run 3 showed that the spline optimiser produced more accurate estimations than the regdiff optimiser, with slightly lower uncertainty bounds as well. Overall however, it appears that the spline optimiser has the most varied results as evident in Figures (c)-(f).

For the regularly cadenced observations, the lowest associated uncertainty is ±0.13 days, corresponding to $\Delta AC = 4.4$ days, with observations made every day. The highest associated uncertainty is ±0.97 days, corresponding to $\Delta CD = 41.5$ days, with an observation made every seventh day (once every week). For the randomly sampled observations, the lowest associated uncertainty is ±0.38 days, corresponding to $\Delta BA = 31.68$ days. The highest associated uncertainty is ±2.23 days, corresponding to $\Delta CD = 45.55$ days. The random observations produced highly unpredictable results with large error bounds associated with each estimation. On the other end, regularly cadenced observations produce results closer to the input delay more regularly, with lower error bounds associated with the estimations.

5 Discussion

There are a variety of lens system for which time delays have been determined. Some two-image gravitational lens systems include B1600+434, PKS 1830-211, B0957+561 and B0218+357, while other systems such as B1608+656 and B1422+231 are quad-image gravitational lens systems. One of the least uncertain of the measured delays comes from B0218+357, with a measured time delay of $\Delta t_{\mathrm{AB}} = 10.5^{+0.4}_{-0.4}$d [38]. The objects in the image of B1600+434 have a estimated delay of $\Delta t_{\mathrm{AB}} = 42.3^{+2.1}_{-2.1}$d [39]. The objects imaged by PKS 1830-211 were found to have a delay of $\Delta t_{\mathrm{AB}} = 26^{+4}_{-5}$d [40]. For the imaged objects of B0957+561, a variety of methods were utilised in determining the time delays between the two objects, with $\Delta t_{\mathrm{AB}} = 459^{+14}_{-16}$d and $\Delta t_{\mathrm{AB}} = 461^{+16}_{-15}$d having the lowest uncertainties [41]. What was used as the basis of the modelled light curves, [10] measured the time delays of B1608+656 to be $\Delta t_{\mathrm{BA}} = 31^{+7}_{-7}$d, $\Delta t_{\mathrm{BC}} = 36^{+7}_{-7}$d and $\Delta t_{\mathrm{BD}} = 76^{+9}_{-10}$d. The time delays of the objects associated with B1422+231 were found to be $\Delta t_{\mathrm{BA}} = 1.5^{+1.4}_{-1.4}$d, $\Delta t_{\mathrm{AC}} = 7.6^{+2.5}_{-2.5}$d and $\Delta t_{\mathrm{BC}} = 8.2^{+2.0}_{-2.0}$d. It is worth noting that many of the observations performed that led to the results above were done using the VLA[22] [42].

Firstly, the comparison between the time delays from [10] and Table 2 reveal the extent to which the SKA can improve on the uncertainties within time delays estimations. As the simulations were based on the lensing system B1608+656, the comparison is particularly compelling. The first point of note is that in the simulated data, the average flux of each object was scaled down by a factor of 10 in comparison to that of [10]. For the sake of convenience, the results of [10] are restated: $\Delta t_{\mathrm{BA}} = 31^{+7}_{-7}$d, $\Delta t_{\mathrm{BC}} = 36^{+7}_{-7}$d and $\Delta t_{\mathrm{BD}} = 76^{+9}_{-10}$d. In direct comparison, the corresponding results of the simulated time delays[23] at a daily cadence were $\Delta t_{\mathrm{BA}} = 31.44^{+0.27}_{-0.27}$d, $\Delta t_{\mathrm{BC}} = 35.84^{+0.25}_{-0.25}$d and $\Delta t_{\mathrm{BD}} = 77.63^{+0.36}_{-0.36}$d. The maximum uncertainty present is 0.36 days, corresponding to 8.64 hours. Furthermore, if one were

[22] The B1830-211 images were observed using the Australia Telescope Compact Array.

[23] The values chosen for comparison were the spline results.

to choose a weekly cadence, the corresponding results were $\Delta t_{\rm BA} = 31.51^{+0.38}_{-0.38}$d, $\Delta t_{\rm BC} = 35.87^{+0.46}_{-0.46}$d and $\Delta t_{\rm BD} = 77.74^{+0.86}_{-0.86}$d. The maximum uncertiatinty present corresponded to 20.64 hours. Although the observation frequency decreased, the uncertainty in the time delays remained below 1 day. The best uncertainty in the time delay results from [10] is 7 days. Thus, even when looking at fainter objects, the SKA's sensitivity is fine enough such that it is able to more accurately detect the objects, resulting is better data from which to construct the light curves with.

Looking at the results from the other lensed images, it appears that much of the uncertainty associated with the determined time delays are on the order of days to weeks. The smallest associated uncertainty, excluding B0218+357, is 1.4 days, whereas the largest associated error is 16 days. Comparing this to the randomised sampling in Table 3, the highest uncertainty present is 3.12 days, found in the Combined Results section. However, many of the uncertainties present within each section of the table lie below 1.5 days. Thus, for poorly sampled data by the SKA, the produced results have uncertainties that are much better than that of the observations made on the other gravitationally lensed systems. This then introduces the idea of using the SKA on these system in the hope that with this kind of sensitivity, the resulting time delays obtained for the lensed images are better in accuracy.

Although there is a lower uncertainty in the measured time delays, PyCS3 appeared to underestimate the time delays. It is important to touch on the possible reason as to why the resulting time delay estimations are further from the input delay than what is expected. As mentioned is Section 4 when discussing the spline fitting, object D, the faintest object, was solely responsible for constructing the end portion of the spline. As a consequence, the portion was incorrectly scaled with respect to the original light curve. The most likely result, based on the data present, is that the mocks generated based on the spline fit were slightly compressed in the time axis. Hence, many of the results were underestimations of what the actual delay was. To combat such a problem, the best solution is to observe over a longer period of time, adjust and align the curves, and from that adjustment, fit a spline to the data were all light curves contribute to the shape. This will likely correct for any outlying points that may dominate in the construction of the spline. To build on this point, the shorter the time delays between the objects are, the shorter the needed period of observation is. Conversely, the longer the time delays between the objects are, the longer the needed period of observation is. Each light curve has the characteristic behaviour of the source object. For shorter time delays, these characteristics are present earlier on than if the time delays were longer.

A shortcoming of the simulations was the absence of a specified ψ[24]. Recall that the Fermat potential requires the deflection potential, ψ. Modelling a deflection potential would have introduced variations that are present within actual lensing systems. The simulations performed were effectively assuming $\psi(t) = C$, with C a constant. A varying deflection potential changes the image position and magnification. As a consequence, the SKA's ability to resolve and detect the objects changes. These changes will reflect in the measured flux and it's associated uncertainty, as well as in the constructed light curves. If ψ varies in such a manner that the SKA's sensitivity and resolution is adequate enough to observe all the objects, then the ability to determine the time delays relies heavily on the software used to do so, along with this how ψ varies (this is touched on soon). If ψ does not magnify the imaged objects enough, or if the objects in the image are too close together to resolve, then the SKA will not accurately measure the fluxes, resulting in the construction of light curves that do fully represent the brightness behaviour of the object. However, if the variability of a lens is too high, then the light-ray travel time from each object changes with a similar variability. The changes in path length experienced by the light rays of the respective objects are necessary related either. Hence, highly variable lenses do not serve much purpose for time-delay cosmography, with more stable lenses being preferred.

With this in mind, recall that $H_0 \propto D_{\Delta t}^{-1}$, where $D_{\Delta t}$ is determined through Equation (20). Referring back, it becomes clear that the deflection potential is required in order to determine $D_{\Delta t}$. As stated, no deflection potential was modelled in the simulation. Hence, $D_{\Delta t}$ could not be determined and subsequently, no H_0 estimation could be made. However, from the obtained results, inferences may be made. From Equation (20), it is clear that the time delay is determined through the time-delay distance. However, it is the time delay that we have. Thus, $D_{\Delta t}$ is determined through the measured time delay. The lesser the uncertainty in the time delay, the better the determination of $D_{\Delta t}$. Subsequently, lower uncertainty in $D_{\Delta t}$ leads to a better estimation of H_0, with a smaller associated error. If a technique such as propagation of error[25] were to be used, then the error assoicated with H_0 would also be considerably

[24] Alternatively, this can also be seen as the absence of a mass model, as both deflection potential and mass model describe the lens.

[25] This technique would be used to convert the uncertainty in $D_{\Delta t}$ to uncertainty in H_0.

lower.

6 Conclusion

The aim of the thesis was to determine to what extent the SKA could better estimate time delays with less uncertainty. With this, the Hubble constant H_0 could then be better estimated. Having simulated a similar setup to the quad-image gravitational lens system B1608+656 with a few parameters adjusted, it was found that in instances where time delays were estimated with uncertainties to the order of days and weeks, the SKA could estimate the delays with uncertainties in the order of hours. In some cases, such as for random observation cadences, the uncertainties grew to days, but by no more than 4 days. Hence, the SKA looks to offer improvement from weeks and days, to hours. With such accuracy, cosmological concepts, such as the Hubble constant may be more accurately determined, and used as an aid in coming to a more conclusive result.

The simulation were idealised however, with many of the real world setbacks unaccounted for. Along with this, no mass model/deflection potential was incorporated into the simulations, simplifying matters even further. However, this does not deny the potential the SKA has in improving the time delays of objects from gravitationally lensed images. Future research could see the incorporation of a deflection potential, with the additional consideration of looking at these objects over multiple frequencies. Hence, the SKA promises to be a great tool for furthering the field of cosmology, providing better data, which in turns, may reveal more about the lineage and future of the Universe.

7 References

References

[1] M. Pettini. *Introduction to Cosmology - Lecture Notes.* Accessed: Day Month Year. n.a. URL: https://people.ast.cam.ac.uk/~pettini/Intro%20Cosmology/.

[2] A. Liddle. *An Introduction to Modern Cosmology.* John Wiley & Sons, Incorporated, 2015. ISBN: 9781118690277.

[3] Sean M. Carro. *Lecture Notes on General Relativity.* arXiv:gr-qc/9712019. Accessed: 2025-10-20. 1997. URL: https://arxiv.org/pdf/gr-qc/9712019.

[4] M. Roos. *Introduction to Cosmology.* John Wiley & Sons, Incorporated, 2015.

[5] E.P. Hubble. "A relation between distance and radial velocity among extra-galactic nebulae". In: *Proceedings of the National Academy of Sciences of the United States of America* 15.3 (1929), pp. 168–173. DOI: 10.1073/pnas.15.3.168.

[6] J.P Hu and F.Y Wang. "Hubble Tension: The Evidence of New Physics". In: *n.a* (2023). arXiv: 2302.05709 [astro-ph.CO]. URL: https://arxiv.org/abs/2302.05709.

[7] S. Birrer et al. "Time-Delay Cosmography: Measuring the Hubble Constant and other cosmological parameters with strong gravitational lensing". In: (2024). arXiv: 2210.10833 [astro-ph.CO]. URL: https://arxiv.org/abs/2210.10833.

[8] NASA. *Hubble Gravitational Lenses.* Accessed on 28 October 2025. NASA. 2025. URL: https://science.nasa.gov/mission/hubble/science/science-behind-the-discoveries/hubble-gravitational-lenses/.

[9] Ta-Pei Cheng. *Relativity, Gravitation and Cosmology : A Basic Introduction.* [ebook] ProQuest Ebook Central. Oxford University Press, 2004. URL: https://ebookcentral-proquest-com.uplib.idm.oclc.org/lib/pretoria-ebooks/detail.action?docID=422865.

[10] C.D. Fassnacht et al. "A Determination of H0 with the Class Gravitational Lens B1608+656: I. Time Delay Measurements with the VLA". In: *The Astrophysical Journal* 527 (1999), pp. 498–512. DOI: 10.1086/308118.

[11] T. Treu. "Strong Lensing by Galaxies". In: (2010). arXiv: 1003.5567 [astro-ph.CO].

[12] Georges Meylan, Philippe Jetzer, and Pierre North, eds. *Gravitational Lensing: Strong, Weak and Micro.* Vol. 33. Saas-Fee Advanced Course. Swiss Society for Astrophysics and Astronomy. Berlin: Springer, 2006. ISBN: 978-3-540-30309-1. DOI: 10.1007/978-3-540-30310-7.

[13] Ramesh Narayan and Matthias Bartelmann. *Lectures on Gravitational Lensing.* arXiv preprint arXiv:astro-ph/9606001v2. Revised version; submitted 3 Jun 1996, last revised 2 Oct 1997. 1997. DOI: 10.48550/arXiv.astro-ph/9606001.

[14] NASA, ESA and L. Calçada. *Gravitational Lensing in Action : Illustration of Gravitational Lensing.* Image credit: NASA, ESA & L. Calçada. Satellite: Hubble Space Telescope. European Space Agency (ESA). Apr. 12, 2011. URL: https://sci.esa.int/web/hubble/-/48616-gravitational-lensing-in-action.

[15] R. Braun et al. *Anticipated Performance of the Square Kilometre Array - Phase 1 (SKA1).* 2019. arXiv: 1912.12699 [astro-ph.IM]. URL: https://arxiv.org/abs/1912.12699.

[16] F. Courbin. "Quasar Lensing: The Observer's Point of View". Version v1. In: (Apr. 28, 2003). arXiv: astro-ph/0304497 [astro-ph]. URL: https://arxiv.org/abs/astro-ph/0304497.

[17] A.R Thompson, J.M Moran, and G.W. Jr Swenson. *Interferometry and Synthesis in Radio Astronomy.* Astronomy and Astrophysics Library. Springer, 2017. ISBN: 978-3-319-44431-4. DOI: 10.1007/978-3-319-44431-4.

[18] J.J. Condor and S.M. Ransom. *Essential Radio Astronomy.* https://www.cv.nrao.edu/~sransom/web/xxx.html. Accessed: 2025-10-21. 2016.

[19] David J. Wilner. *Imaging and Deconvolution.* Lecture at the 2021 SMA School, Centre for Astrophysics, Harvard & Smithsonian; video available at https://www.youtube.com/watch?v=c-2YH02_yN0&list=WL&index=2&t=146s. Accessed: 22 October 2025; key point at 32:23. 2021.

[20] G. B. Taylor, C. L. Carilli, and R. A. Perley, eds. *Synthesis Imaging in Radio Astronomy II.* Vol. 180. ASP Conference Series. Proceedings of the Sixth NRAO/NMIMT Synthesis Imaging Summer School, Socorro, NM, June 17–23, 1998. National Radio Astronomy Observatory. 1999.

[21] T.L. Wilson, K. Rohlfs, and S. Hüttemeister. *Tools of Radio Astronomy.* Fifth. Springer, 2009. ISBN: 978-3-540-85121-9. DOI: 10.1007/978-3-540-85122-6.

[22] J.A. Högbom. "Aperture Synthesis with a Non-regular Distribution of Interferometer Baselines". In: *Astronomy and Astrophysics Supplement, p.417* 15 (1974), pp. 417–426.

[23] ALMA Video Working Group. *ALMA Primer Series: Introduction to CLEAN.* YouTube video, accessed on 22 October 2025. NRAO–NAASC and NRC–HAA. 2022. URL: https://www.youtube.com/watch?v=fF3KetPUyFE.

[24] C.D. Fassnacht et al. "A Determination of H0 with the Class Gravitational Lens B1608+656: III. A Significant Improvement in the Precision of the Time Delay Measurements". In: *The Astrophysical Journal* 581 (2002), pp. 823–835. DOI: https://doi.org/10.1086/344368.

[25] The CASA Team et al. "CASA, the Common Astronomy Software Applications for Radio Astronomy". In: *Publications of the Astronomical Society of the Pacific* 134 (2022), p. 114501. DOI: 10.1088/1538-3873/ac9642.

[26] *CASA Documentation: casatools.* NRAO. 2021. URL: https://casadocs.readthedocs.io/en/stable/api/casatools.html.

[27] *CASA Documentation: casatasks.* NRAO. 2021. URL: https://casadocs.readthedocs.io/en/stable/api/casatasks.html#.

[28] *SKAO Sensitivity Calculator.* SKAO. 2025. URL: https://sensitivity-calculator.skao.int.

[29] J. A. Ott and the CARTA Team. "CARTA: Cube Analysis and Rendering Tool for Astronomy". In: *Bulletin of the American Astronomical Society* 52.1 (2020). American Astronomical Society Meeting #235, p. 364.11. DOI: 10.5281/zenodo.3377984.

[30] Astropy Collaboration et al. "Astropy: A community Python package for astronomy". In: *aap* 558, A33 (Oct. 2013), A33. DOI: 10.1051/0004-6361/201322068. arXiv: 1307.6212 [astro-ph.IM].

[31] A. M. Price-Whelan et al. "The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package". In: *aj* 156, 123 (Sept. 2018), p. 123. DOI: 10.3847/1538-3881/aabc4f.

[32] M. Tewes, F. Courbin, and G. Meylan. "COSMOGRAIL XI: Techniques for time delay measurement in presence of microlensing". In: *Astronomy & Astrophysics* 553 A120 (2013). DOI: https://doi.org/10.1051/0004-6361/201220123.

[33] M. Millon et al. "PyCS3: A Python toolbox for time-delay measurements in lensed quasars". In: *The Journal of Open Source Software* 5.53 (2020), p. 2654. DOI: 10.21105/joss.02654.

[34] Martin Millon, Vivien Bonvin, and Malte Tewes. *PyCS3 Documentation.* https://cosmograil.gitlab.io/PyCS3/apidoc/pycs3.html. Accessed: 2025-01-23. 2020.

[35] J.B. Oke. "Absolute Spectral Energy Distributions for White Dwarfs". In: *The Astrophysical Journal Supplement Series* 27 (1974), pp. 21–35.

[36] P.R. Bevington and D.K. Robinson. *Data Reduction and Error Analysis for the Physical Sciences Third Edition.* McGraw-Hill, 2003. ISBN: 0-07-247227-8.

[37] U.M. Ascher and C. Grief. *A First Course in Numerical Methods.* Society for Industrial and Applied Mathematics, 2011. ISBN: 978-0-89871-998-7. DOI: https://doi.org/10.1137/9780898719987.

[38] A. D. Biggs et al. “Time delay for the gravitational lens system B0218+357”. Version v1. In: (Nov. 18, 1998). DOI: 10.1046/j.1365-8711.1999.02309.x. arXiv: astroph/9811282 [astro-ph]. URL: https://arxiv.org/abs/astro-ph/9811282.

[39] A. D. Biggs. “The time delay of CLASS B1600+434 from VLA multi-frequency and polarization monitoring”. Version v1. In: (May 17, 2021). DOI: 10.1093/mnras/stab1444. arXiv: 2105.07905 [astro-ph.GA]. URL: https://arxiv.org/abs/2105.07905.

[40] J. E. J. Lovell et al. “The Time Delay in the Gravitational Lens PKS 1830-211”. Version v1. In: (Sept. 24, 1998). arXiv: astro-ph/9809301 [astro-ph]. URL: https://arxiv.org/abs/astro-ph/9809301v1.

[41] D.B. Haarsma et al. “The 6cm Light Curves of B0957+561, 1979-1994: New Features and Implications for the Time Delay”. Version v2. In: (Nov. 26, 1996). DOI: 10.1086/303860. arXiv: astroph/9607080 [astro-ph]. URL: https://arxiv.org/abs/astro-ph/9607080v2.

[42] A.R. Patnaik and D. Narasimha. “Determination of time delay from the gravitational lens B1422+231”. In: *Monthly Notices of the Royal Astronomical Society* (2001). DOI: 10.1111/j.1365-2966.2001.04711.x. arXiv: astro-ph/0106104 [astro-ph].

A Simulated Light Curve Code

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker

plt.rcParams['font.family'] = 'STIXGeneral'
plt.rcParams['mathtext.fontset'] = 'stix'
plt.rcParams['font.size'] = 23
plt.rcParams['axes.labelsize'] = 23
plt.rcParams['axes.titlesize'] = 23
plt.rcParams['xtick.labelsize'] = 23
plt.rcParams['ytick.labelsize'] = 23
plt.rcParams['legend.fontsize'] = 16

#These are the parameters to model the master light curve
Par1 = [2, 3, 2.1, 0]
Par2 = [-1, 6.2, -3, 0]
Par3 = [3, -1, -2.6, 0]
Par4 = [0.6, 9.2, 0.8, 0]
Pars = [Par1, Par2, Par3, Par4]

#This produces a normalized curve
def Intensity_signal(Parameters, x_value):
    Comp = 0
    for i in range(0, len(Parameters)):
        Comp += Parameters[i][0]*np.sin(Parameters[i][1]*(x_value) + Parameters[i][2]) + Parameters[i][3]
    PT_d = 0.08/(max(Comp) - min(Comp)) #PT_d is peak-trough distance
    Normalized_shift = 1 - np.average(PT_d*Comp) #Shifts the data such that it's average is about 1
    return PT_d*Comp + Normalized_shift

#Since we are working with sinusoidally constructed curves, we let 2pi be equivalent to 442 segments
#That values comes from the max time delay of 77.0 days, plus 365 days (to simulates a full year of daily observations)
Ave_fluxes = np.array([34.1e-4, 16.8e-4, 17.3e-4, 5.90e-4])
time_delays = [31.5, 0, 36.2, 77.9]
t_vals = np.linspace(1,366, 365)

#This is the radian conversion of the time delays
segment_size = (2*np.pi)/442.9
base = 365*segment_size
time_delays_rad = segment_size*np.array(time_delays)
```

```

plt.figure(figsize=(10,6))
plt.xlabel("Time (days)")
plt.ylabel("Flux Density (Jy)")
Master_lg_domain = np.linspace(0, 2*np.pi, 443)
Master_lg_x_axis = np.linspace(1, 443, 443)
Master_light_curve = Intensity_signal(Pars, Master_lg_domain)
plt.plot(Master_lg_x_axis, Master_light_curve, color = ’black’, linewidth = 1)
plt.grid(True)
plt.savefig(’Master Curve.png’, dpi = 600,bbox_inches=’tight’)
plt.show()

# Contructing the intervals with the appropriate time delays
# Curve A interval
A_interval = [time_delays_rad[0], base + time_delays_rad[0]]
A_int_vals = np.linspace(A_interval[0], A_interval[1], 365)
# Curve B interval
B_interval = [time_delays_rad[1], base + time_delays_rad[1]]
B_int_vals = np.linspace(B_interval[0], B_interval[1], 365)
# Curve C interval
C_interval = [time_delays_rad[2], base + time_delays_rad[2]]
C_int_vals = np.linspace(C_interval[0], C_interval[1], 365)
# Curve D interval
D_interval = [time_delays_rad[3], base + time_delays_rad[3]]
D_int_vals = np.linspace(D_interval[0], D_interval[1], 365)


# All the light curves with the appropriate delays
Curve_A = Ave_fluxes[0]*Intensity_signal(Pars,A_int_vals)
Curve_B = Ave_fluxes[1]*Intensity_signal(Pars,B_int_vals)
Curve_C = Ave_fluxes[2]*Intensity_signal(Pars,C_int_vals)
Curve_D = Ave_fluxes[3]*Intensity_signal(Pars,D_int_vals)
Flux_Values = [Curve_A, Curve_B, Curve_C, Curve_D]

# Writing the fluxes to a file called Flux_Values
with open(’Flux_Values.txt’, ’w’) as file:
    for i in range(0, 365):
        if(i<364):
            String = str(Flux_Values[0][i]) + ’ ’ + str(Flux_Values[1][i]) + ’ ’
                 + str(Flux_Values[2][i]) + ’ ’ + str(Flux_Values[3][i]) + ’\n’
            file.write(String)
        else:
            String = str(Flux_Values[0][i]) + ’ ’ + str(Flux_Values[1][i]) + ’ ’
                 + str(Flux_Values[2][i]) + ’ ’ + str(Flux_Values[3][i])
            file.write(String)

# Plotting of the above light curves
plt.figure(figsize=(10,6))
plt.xlabel("Time (days)")
plt.ylabel("Flux Density (mJy)")
plt.plot(t_vals, Curve_A, label = ’A’, color = ’green’, linewidth = 1)
plt.plot(t_vals, Curve_B, label = ’B’, color = ’blue’, linewidth = 1)
plt.plot(t_vals, Curve_C, label = ’C’, color = ’red’, linewidth = 1)
plt.plot(t_vals, Curve_D, label = ’D’, color = ’orange’, linewidth = 1)
ax = plt.gca()
ax.yaxis.set_major_formatter(ticker.FuncFormatter(lambda y, _: f"{y*1000:.1f}"))
plt.legend(loc=’upper right’, bbox_to_anchor=(1, 0.8))
plt.grid(True)
plt.savefig(’Light Curves.png’, dpi = 600,bbox_inches=’tight’)
plt.show()
```

B CASA Simulation Code

```
import numpy as np
import os
import random as rd

Flux_vals = np.loadtxt('../Flux_Values.txt', comments = '#', usecols = (0,1,2,3))
Dates_1 = []
Dates_2 = []
with open('../String_Dates_1.txt', 'r') as file:
    for line in file:
        Dates_1.append(line.replace('\n', ''))
with open('../String_Dates_2.txt', 'r') as file:
    for line in file:
        Dates_2.append(line.replace('\n', ''))

dir_1 = 'J2000 00.00.00.0 -45.00.00.0'
dir_2 = 'J2000 -00.00.00.738 -45.00.01.962'
dir_3 = 'J2000 -00.00.00.745 -45.00.00.453'
dir_4 = 'J2000 00.00.01.130 -45.00.01.257'

directions = [dir_1, dir_2, dir_3, dir_4]
for i in range(0, len(Flux_vals)):
    os.mkdir(Dates_2[i])
    os.chdir(Dates_2[i])
    for j in range(0, 4):
        cl.addcomponent(
            flux = Flux_vals[i][j],
            fluxunit = 'Jy',
            dir = directions[j],
            freq = '6.55GHz',
            shape = 'point'
        )
    fname = f'{Dates_2[i]}_complist.cl'
    cl.rename(filename = fname)
    cl.done()

    vp.reset()
    vp.setpbairy(telescope='SKA_MID', dishdiam=15, maxrad='1deg', reffreq='6.55GHz', dopb = True, blockagediam = 0.0)

    proj_name = f'Obs_data_{Dates_2[i]}'

    simobserve(
        project = proj_name,
        refdate = Dates_1[i],
        complist = fname,
        compwidth = '3.9GHz',
        setpointings = True,
        integration = '45s',
        obsmode = 'int',
        thermalnoise = '',
        antennalist = '../SKA_MID_15M_AA4.txt',
        hourangle = 'transit',
        totaltime = '45s',
        graphics = 'none')

    source = f'{proj_name}/{proj_name}.SKA_MID_15M_AA4.txt.ms'
    dest = f'Noisy_{Dates_2[i]}.ms'
    os.system(f'cp -r {source} {dest}')
    sm.openfromms(dest)
    sigma_simple = '0.5558mJy'
    sm.setnoise(mode = 'simplenoise', simplenoise = sigma_simple)
    sm.corrupt()
    sm.done()

    im_name = 'obs_'+ Dates_2[i]

    tclean(
    vis = dest,
    datacolumn ='data',
    imagename = im_name,
    imsize = 2000,
    cell = '0.0030182188745507033arcsec',
    specmode = 'mfs',
    gridder = 'standard',
    deconvolver = 'hogbom',
    weighting = 'briggs',
    robust = -0.5,
    mask = '../mask.image',
    threshold = '25.85uJy',
    niter = 20000
    )
    pix_pos_1 = '959, 1284, 999, 1324'
    pix_pos_2 = '1132, 634, 1172, 673'
    pix_pos_3 = '1134, 1134, 1174, 1174'
    pix_pos_4 = '695, 868, 735, 908'
    pix_pos = [pix_pos_1, pix_pos_2, pix_pos_3, pix_pos_4]


    # This reads the flux of each object into a file called Measured_flux.txt
    # Each date folder contains one of these files
    with open('Measured_flux.txt', 'w') as file:
        for i in range(0, len(pix_pos)):
            result = imfit(imagename = f'{im_name}.image', box = pix_pos[i])
            peak_val = result['results']['component0']['flux']['value'][0]
            peak_error = result['results']['component0']['flux']['error'][0]
            if(i != len(pix_pos) - 1):
                String = str(peak_val) + ' ' + str(peak_error) + '\n'
            else:
                String = str(peak_val) + ' ' + str (peak_error)
            file.write(String)
    os.chdir('../')
```

C Flux Harvesting and Conversion Code

```
# This file reads all of the Measured_flux.txt file withing each date folder
# This file is to be run in the 'Total Observations' folder
import numpy as np
import os
import math

# This will allow us to cd into each folder
Dates = []
with open('../String_Dates_2.txt', 'r') as file:
    for line in file:
        Dates.append(line.replace('\n', ''))
days_365 = [i for i in range(1, 366)]

#To convert the measured fluxes to magnitudes
def mag_convert(flux_val, flux_err, val):
    if(val == 0):
        return str(-2.5*math.log10(float(flux_val)/3631))
    else:
        return str((2.5/math.log(10))*(float(flux_err)/float(flux_val)))

```

```
# This is for PyCS3
JD = 2460677.0 - 2400000 - 1

for i in range(0, len(Dates)):
    os.chdir(Dates[i])

    String_data = str(JD + days_365[i]) + ’\t’
    String_data_mag = str(JD + days_365[i]) + ’\t’

    with open(’Measured_flux.txt’, ’r’) as file:
        lines = file.readlines()

    for i, line in enumerate(lines):
        Flux_vals = line.split(" ")
        if i == len(lines) - 1:
            String_data += Flux_vals[0] + ’\t’ + Flux_vals[1] + ’\n’
            String_data_mag += mag_convert(Flux_vals[0], Flux_vals[1], 0) + ’\t’
            String_data_mag += mag_convert(Flux_vals[0], Flux_vals[1], 1) + ’\n’
        else:
            String_data += Flux_vals[0] + ’\t’ + Flux_vals[1].replace(’\n’, ’’)
                + ’\t’
            String_data_mag += mag_convert(Flux_vals[0], Flux_vals[1], 0) + ’\t’
            String_data_mag += mag_convert(Flux_vals[0], Flux_vals[1], 1) + ’\t’

    with open(’../Flux_readings.txt’, ’a’) as file:
        file.write(String_data)
    with open(’../Magnitude_readings.txt’, ’a’) as file:
        file.write(String_data_mag)
    os.chdir(’..’)
```

D PyCS3 Code

```
import pycs3.gen.lc
import pycs3.gen.lc_func
import pycs3.gen.mrg
import pycs3.sim.draw
import pycs3.sim.run
import pycs3.sim.plot
import pycs3.tdcomb
import pycs3.tdcomb.comb
import pycs3.tdcomb.plot
import pycs3.regdiff.multiopt
import myopt
import time
import pickle
import numpy as np
from multiprocess import Pool

filename = ’data/Magnitude_readings.txt’
lcs = [
    pycs3.gen.lc_func.rdbimport(filename, ’B’, ’mag_B’, ’magerr_B’, "SKA"),
    pycs3.gen.lc_func.rdbimport(filename, ’A’, ’mag_A’, ’magerr_A’, "SKA"),
    pycs3.gen.lc_func.rdbimport(filename, ’C’, ’mag_C’, ’magerr_C’, "SKA"),
    pycs3.gen.lc_func.rdbimport(filename, ’D’, ’mag_D’, ’magerr_D’, "SKA")
]
# The light curves have been captured
# The below code colourises the light curves
pycs3.gen.mrg.colourise(lcs)

lcs[0].shifttime(0)
lcs[1].shifttime(31.5)
lcs[2].shifttime(36.2)
```

```
lcs[3].shifttime(77.9)

pycs3.gen.util.writepickle(lcs, ’data/pure_lcs.pkl’)
spline = myopt.spl(lcs)
pycs3.gen.util.writepickle((lcs, spline), ’data/optspl.pkl’)
pycs3.gen.lc_func.display(lcs, [spline])
pycs3.sim.draw.saveresiduals(lcs, spline)

pycs3.sim.draw.multidraw(lcs,
    onlycopy=True,
    n=50,
    npkl=10,
    simset="copies")

pycs3.sim.draw.multidraw(lcs,
    spline,
    onlycopy=False,
    n=50,
    npkl=10,
    simset="sim1tsr10",
    shotnoise="mcres",
    shotnoisefrac=1.0,
    truetsr=10.0,
    tweakml={},
    tweakspl=None)

lcs = pycs3.gen.util.readpickle(’data/pure_lcs.pkl’)
def spl_multirun_call(args):
    return spl_multirun(*args)
def spl_multirun(i, chosen_file, lcs, myopt_func, optset, tsrand, destpath):
    results = pycs3.sim.run.multirun(chosen_file,
        lcs,
        myopt_func,
        optset=optset,
        tsrand=tsrand,
        kwargs_optim={},
        keepopt=True,
        destpath=destpath)
    return results

lcs = pycs3.gen.util.readpickle(’data/pure_lcs.pkl’)
def regdiff_multirun_call(args):
    return regdiff_multirun(*args)
def regdiff_multirun(i, at_name, lcs, myopt_func, optset, tsrand):
    result = pycs3.sim.run.multirun(at_name,
        lcs,
        myopt_func,
        optset=optset,
        tsrand=tsrand,
        kwargs_optim = {},
        keepopt=True,
        destpath = ’./’)
    return result

nworkers = 8

argument1_spl = [(j, "copies", lcs, myopt.spl, ’spl1’, 10.0, "./") for j in
    range(nworkers)]
argument2_spl = [(j, "sim1tsr10", lcs, myopt.spl, ’spl1’, 10.0, "./") for j in
    range(nworkers)]
argument1_regdiff = [(j, ’copies’, lcs, myopt.regdiff,’regdiff’,10.0) for j in
    range(nworkers)]
```

```python
argument2_regdiff = [(j, 'sim1tsr10', lcs, myopt.regdiff,'regdiff',10.0) for j in range(nworkers)]

pool_val = Pool(nworkers)
spl_copies = pool_val.map(spl_multirun_call, argument1_spl)
pool_val = Pool(nworkers)
spl_sim1 = pool_val.map(spl_multirun_call, argument2_spl)

pool_val = Pool(nworkers)
result_list_copies_regdiff = pool_val.map(regdiff_multirun_call, argument1_regdiff)
pool_val = Pool(nworkers)
result_list_sim1_regdiff = pool_val.map(regdiff_multirun_call, argument2_regdiff)

#Reading from the regdiff files
with open('sims_copies_opt_regdiff_delays.pkl', 'rb') as f:
    rg_delays = pickle.load(f)
with open('sims_sim1tsr10_opt_regdiff_errorbars.pkl', 'rb') as f:
    rg_errors = pickle.load(f)
#Reading from the spline files
with open('sims_copies_opt_spl1_delays.pkl', 'rb') as f:
    sp_delays = pickle.load(f)
with open('sims_sim1tsr10_opt_spl1_errorbars.pkl', 'rb') as f:
    sp_errors = pickle.load(f)

combined = pycs3.tdcomb.comb.combine_estimates(group_list, sigmathresh=0.0, testmode=False)
combined.linearize(testmode=False)

combined_delays = combined.medians
combined_errors_up = combined.errors_up
combined_errors_down = combined.errors_down

#Saving the data to a file called regdiff_data.txt
with open('regdiff_data.txt', 'w') as file:
    for i in range(0, 6):
        String = rg_delays.data[i]['label'] + '\t' + str(rg_delays.data[i]['mean']) + '\t' + str(rg_errors.data[i]['tot']) + '\n'
        file.write(String)
#Saving the data to a file called spline_data.txt
with open('spline_data.txt', 'w') as file:
    for i in range(0, 6):
        String = sp_delays.data[i]['label'] + '\t' + str(sp_delays.data[i]['mean']) + '\t' + str(sp_errors.data[i]['tot']) + '\n'
        file.write(String)
#Saving the data to a file called combined_data.txt
with open('combined_data.txt', 'w') as file:
    for i in range(0, 6):
        String = sp_delays.data[i]['label'] + '\t' + str(combined.medians[i]) + '\t' + str(combined.errors_up[i]) + '\t' + str(combined.errors_down[i]) + '\n'
        file.write(String)
```