Maxime Rey
Supervised by Yannick Copin

Institut de Physique Nucléaire de Lyon
February, 25th - June, 24th

Modelisation of slitless spectroscopic observations from the Auxiliary Telescope within the framework of the Large Synaptic Survey Telescope

Abstract

In 2022, the LSST will begin its observation and one of its goals is to study dark energy further. However, to do this efficiently, ten times its actual precision is needed. To be able to reach it, the atmosphere must be characterised more precisely and this is why an Auxiliary Telescope was added. Its only purpose is to measure the atmospheric parameters and to do this, it will use an end-to-end forward modelling comprised of two parts. The effect of the atmosphere on the spectrum of a known star will be modelled to simulate an observed spectrum and at the same time, the Auxtel will observe this star to produce a spectrum. By comparing these two and matching them, the atmospheric parameters are going to be inferred. In those models, at least 3 effects had to be modelled: the defocus of the point spread function, the atmospheric turbulence and the atmospheric differential refraction. These are at the heart of this work. I modelled the defocus using two libraries (HCIPy and POPPy) and two methods (modelling a lens and using the Zernike polynomials). In a first time I tried to model a lens in HCIPy but it proved inconclusive. Then, I used Zernike polynomials to model the effect of the defocus with HCIPy. In POPPy, a function was implemented to model a lens but it was in reality also using Zernike polynomials. By comparing these two libraries, it was found that they were not inducing a similar defocus. The effect of defocus was greatly stronger for POPPy than for HCIPy. Furthermore, they were acting on different ranges of defocus. These made it hard to conclude which one was the more adapted and if they were correct. A potential solution to this would be to model a lens in Fourier space without using libraries. To model the atmospheric turbulence, I first used HCIPy and Soapy. These gave coherent results but took a consequent computational time and were deemed not adapted. Therefore, I used 2 models to exploit the optical transfer function of Kolmogorov and compared them. The results were in good agreement and these models were consequently faster so this result was conclusive. The atmospheric differential refraction is described analytically and there was already a script using these equations. I adapted it to the current case so that it would automatically output the value of the atmospheric differential refraction for an observation. It worked and matched results previously computed [1] and is currently being implemented in a library developed for the Auxtel called Spectractor.

Master 2 - Magistère 3 internship
Université de Lyon - Université Claude Bernard Lyon 1
Magistère de Physique Fondamentale d'Orsay

Contents

I Introduction 5

1 Large Synoptic Survey Telescope 5
- 1.1 Specifications 5
- 1.2 Goals 5
- 1.3 Dark Energy Science Collaboration (DESC) 6

2 The Auxiliary Telescope (Auxtel) 6
- 2.1 Goal 6
- 2.2 Operating method 7

3 The dispersion method 8
- 3.1 Dispersion with slitless spectroscopy 8
- 3.2 The Ronchi grating 8

4 Point spread function (PSF) 9
- 4.1 Principle 9
- 4.2 Full width half maximum (FWHM) 9

5 Atmospheric turbulence 10
- 5.1 Kolmogorov model 10
- 5.2 Extension to Kolmogorov's model 10

6 Fourier optics 10

7 Mathematical description of the image observed 11

II First steps of the model 11

1 Tools for the models 11

2 Optical propagation 12
- 2.1 Propagation to the focal plane 12
- 2.2 Fresnel and Fraunhofer diffraction 12

3 Aperture simulation 13
- 3.1 Shape of the aperture 13
- 3.2 HCIPy 13
- 3.3 POPPy 14

4 Comparison of the diffraction PSF from HCIPy and POPPy 15

III Defocus 16

1 Theory to model the defocus 16
- 1.1 Lens implementation 16
- 1.2 Defocus with Zernike polynomials 16
  - 1.2.1 Third Zernike polynomial 16
  - 1.2.2 Sagitta difference 17
  - 1.2.3 Application 18

2 Model of a lens with HCIPy 18
- 2.1 Consistency check 18
- 2.2 Constraints for defocus 18
- 2.3 Adding the defocus 18
- 2.4 Zernike polynomials with HCIPy 19

3 POPPy **20**

4 Comparison between the defocus of HCIPy and POPPy **21**

4.1 Discrepancy 21

4.2 Limits 22

4.2.1 POPPy 22

4.2.2 HCIPy 22

4.3 Solution 23

IV Atmosphere 23

1 Atmospheric turbulence (seeing) **24**

1.1 HCIPy 24

1.2 Soapy 25

1.3 Using the optical transfer function 25

1.3.1 Overview 25

1.3.2 Comparison of the four computations 26

1.3.3 Comparison with HCIPy 26

1.3.4 Implementation 27

2 Atmospheric differential refraction **27**

2.1 Phenomenon 27

2.2 Mathematical description 28

2.3 Modelling 28

3 Noises **29**

Conclusion 30

Appendix 31

The code **31**

. 31

Automation of the codes 31

Examples of utilisation 31

Bibliography 33

List of Figures

1 Artist view of the LSST and Auxtel . . . 5
2 LSST z-passband . . . 6
3 Impact of the transmission of the atmosphere on the LSST filters . . . 7
4 Operating sketch of Auxtel . . . 7
5 Example of slitless spectroscopy . . . 8
6 Gratings considered for the Auxtel. The images are from a LSST conference. . . . 8
7 Example of the effect of the PSF . . . 9
9 Diffraction pattern for a circular and a square aperture. Both images are from Wikipedia. . . . 13
10 Modelling of the PSF of the aperture of Auxtel with HCIPy. . . . 14
11 Modelling of the PSF of the aperture of Auxtel with POPPy. . . . 14
12 Sum and slice of the PSF . . . 15
13 Re-scaled sum and slice of the PSF . . . 15
14 Re-scaled sum and slice of the PSF with low sampling . . . 16
15 Sketch of the sagittas difference . . . 17
16 Sketch picturing the different notations. . . . 18
17 Image of the focus and defocused PSF using HCIPy . . . 19
18 Comparison of HCIPy plots. . . . 20
19 Image of the focus and defocused PSF . . . 20
20 Comparison of HCIPy plots. . . . 21
23 Comparison of various defocus with POPPY. . . . 22
24 Comparison of various defocus with HCIPy. . . . 23
26 Result of a realisation of the atmosphere on the PSF using HCIPy . . . 24
27 Result of 100 realisations of the atmosphere averaged on the PSF using HCIPy. . . . 24
28 Slice and sum of the PSF without and with atmosphere using POPPy . . . 25
29 Slice of the PSF without and with atmosphere using HCIPy . . . 25
30 Comparison of 4 Kolmogorov computations methods. . . . 26
33 Atmospheric differential refraction . . . 28
34 Example of the effect of the ADR . . . 29

List of Tables

1 Comparison of the python libraries . . . 11
2 ADR statistics . . . 29

Part I

Introduction

1 Large Synoptic Survey Telescope

1.1 Specifications

In 2022, a new telescope called the Large Synoptic Survey Telescope (LSST) shown in figure 1, will begin operating for a survey of at least 10 years. This earth telescope with a primary mirror of 8.4m is expected to provide unprecedented data for astrophysics and cosmology. Unlike previous telescopes, it has an extremely

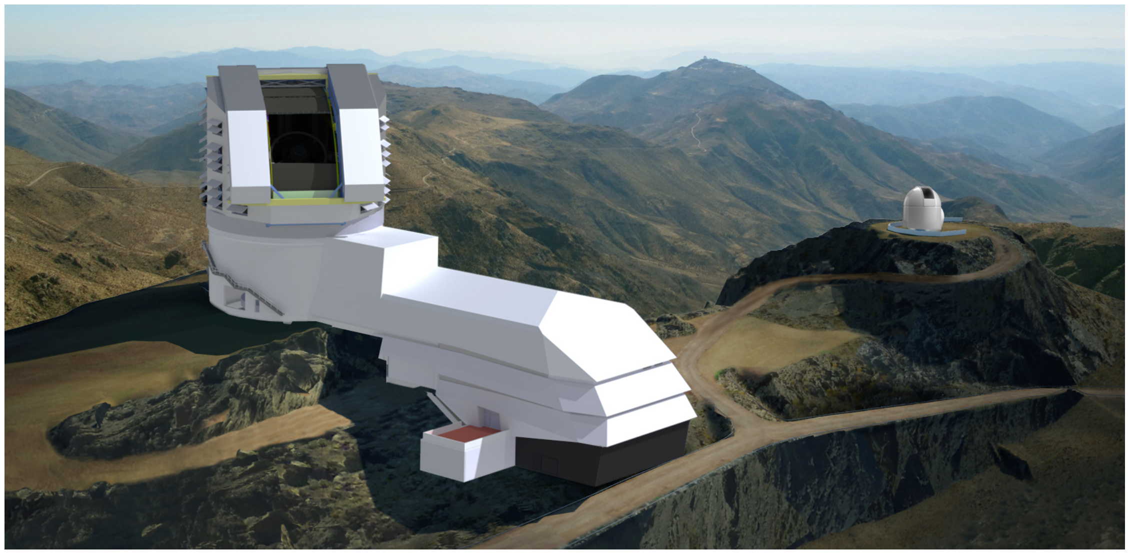


Figure 1: Renderings of both the LSST (on the left) and the Auxtel (on the right). The Auxtel will be used to measure the transmission of the atmosphere to improve the precision of the LSST to 1 mmag. Image from https://www.lsst.org/about/tel-site/

large field of view -approximately 10 square degrees- which will allow it to survey the entirety of the austral sky approximately every 3 nights and with a collecting area of forty square meters. Its camera will be the largest ever constructed, its size being 1.65m $\times$ 3m, with a resolution of 3.2 Gigapixels which will be producing approximately 20 Terabytes of raw data every night. With its 5 filters (g, r, i, z and y) it will be able to see a spectral range from 320nm to 1050nm.

1.2 Goals

Four main goals were defined for the LSST. The first one is the study of dark energy and dark matter which were implemented in the cosmological model due to a lack of understanding concerning certain phenomena. Dark energy and dark matter were introduced respectively to justify the accelerated expansion of the universe and gravitational effects on galaxies but are both still obscure and only hypothesised. As a second goal, the LSST will look at the solar system in more depth and will track and analyse near-Earth objects (NEOs) situated as far as in the Main Belt. For example, it is expected to detect between 60% and 90% of potentially hazardous asteroids (PHAs) larger than 140m in diameter. All these will also give insight into the history of our solar system through the analysis of the compositions of such objects. Thirdly, by seeing the same place several times through its course, an evolution in time of the universe will be acquired which will allow the study of variables phenomenon such as supernovae with unparalleled precision. If a transient is detected, LSST will signal it so it can be further studied before it fades away. Following a similar idea are the standard sirens: if a gravitational wave is detected, the LSST will be used to pinpoint its source and analyse it further as we cannot access the redshift of the source with gravitational waves alone. The last goal defined for the LSST concerns the Milky Way and particularly its stars as they offer an opportunity to obtain a better understanding of its history.

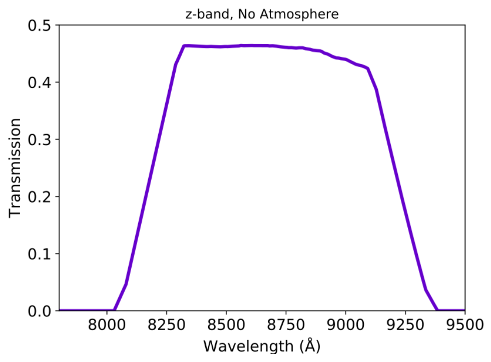
1.3 Dark Energy Science Collaboration (DESC)

One of the most interesting topics in today's cosmology concerns the accelerated expansion of the universe. The cause of it has been coined as dark energy but, as its name suggests, we have little knowledge of it. The Dark Energy Science Collaboration (DESC) will play a major part regarding this topic with LSST. The collaboration is already preparing for the incoming telescope. The principal topics of the collaboration concern both the strong and weak gravitational lensing, the large-scale structures, type Ia supernovae and galaxy clusters. However, to really gain something with the LSST using supernovae, a precision in magnitude of 10^{-3} must be achieved but the LSST currently aims for 10mmag. To improve by a factor 10 the precision of the LSST, an Auxiliary telescope (Auxtel) will be added next to the LSST as shown in figure 1. It will measure the transmission of the atmosphere for all LSST bandpass. The underlying goal of this is to correct the error induced by the atmosphere on the images. The Auxtel, previously known as Calypso telescope and situated on Kitt Peak was donated to LSST in 2008 by Dr Edgar Smith and is a telescope with a primary dish of 1.2 meters and an f-number (the ratio of the focal over the dish diameter) of 18.

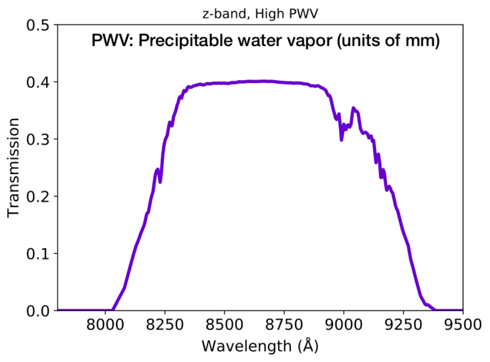
2 The Auxiliary Telescope (Auxtel)

2.1 Goal

To reach a precision of 0.1% and be able to return the most precise measurements possible, an Auxiliary telescope was added near the LSST. This telescope is called Auxtel and its goal is to monitor the atmospheric transmission along the line of sight. To understand how it is useful, it is important to grasp how filters are used within telescopes. A filter is an optical element which limits the observation to a certain range of wavelengths. This consequently improves contrast with the background (and its noises from other wavelengths) which enhances the images. However, in reality, the filter considered is called the passband and is comprised of the filter plus the effects of the instrument and the atmosphere. To get a good calibration of the fluxes, this passband must be described very precisely so that even though the atmosphere changes with time, the flux assigned to a certain object will not be influenced by this effect. An example of consequent alteration in the flux received due to the atmosphere can be shown by considering water vapour. Its effect on the LSST z filter is shown on figure 2.



(a) LSST z-passband without atmosphere.



(b) LSST z-passband with atmosphere.

Figure 2: LSST z-passband without (left) and with (right) the impact of the water vapour in the atmosphere.

In reality, there are other components of the atmosphere which will impact these filters even further. Along with water vapour, the main ones are ozone, aerosols and Rayleigh scattering. As the LSST is equipped with 5 filters, they all need to be characterised with very high precision. Furthermore, the difference in the transmission is not constant over all filters as the transmission of the atmosphere depends on the wavelength considered as shown in figure 3.

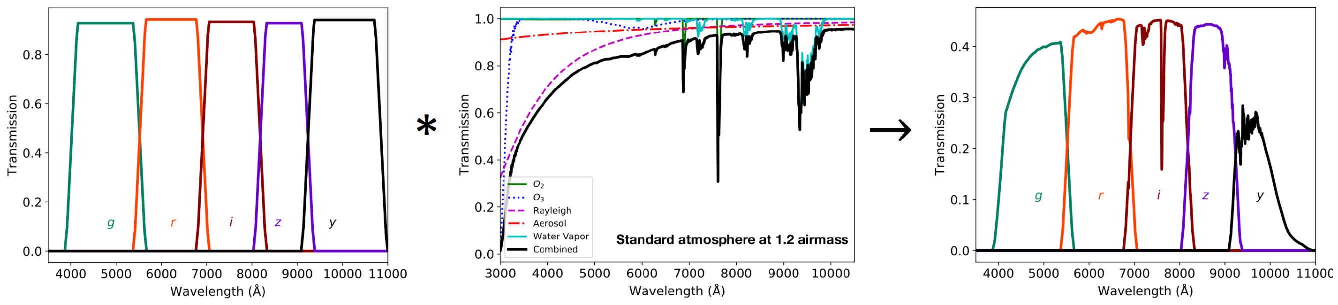


Figure 3: Impact of the transmission of the atmosphere on the LSST filters. On the left is the initial transmission of the filters which are then influenced by the atmosphere shown in the centre which results in the passbands shown on the right. The images are from a DESC conference.

Characterising this alteration due to the atmosphere for each filter is the purpose of the Auxtel.

2.2 Operating method

To infer the parameters describing the atmosphere, two branches have to be considered as shown in figure 4.

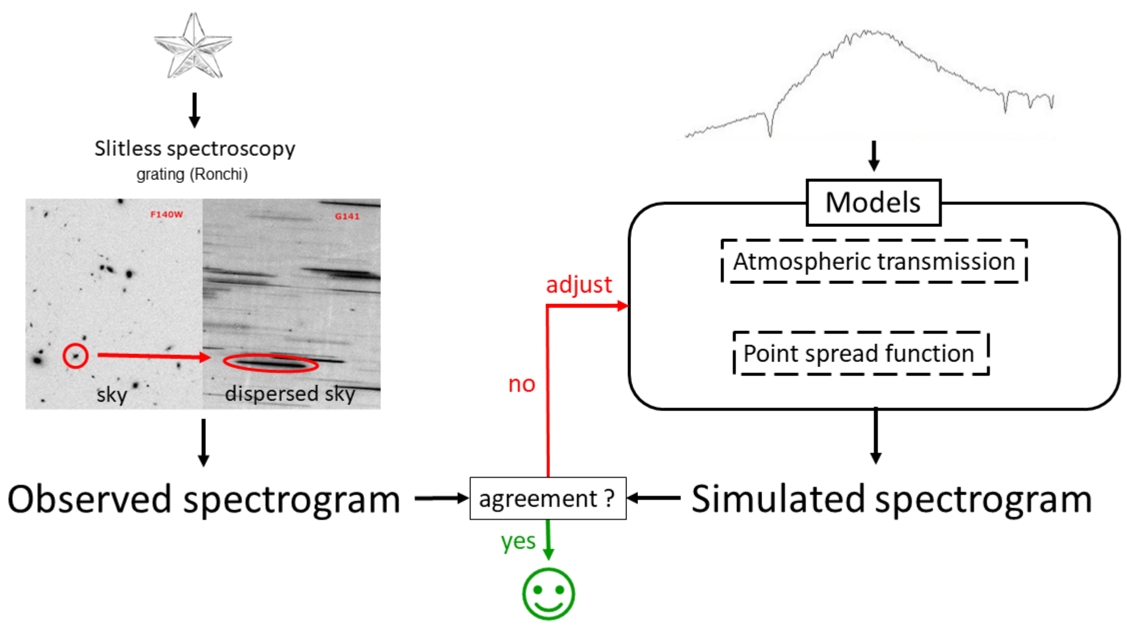


Figure 4: Sketch explaining roughly the mode of operation of the Auxtel. On the left is shown how the science image is obtained and on the right is shown how the simulated spectrogram is obtained. Then, both are compared and depending on the result, the right branch is iterated. The images are from a DESC conference.

The Auxtel observes a known star which it disperses with a grating. This gives a two-dimensional spectrum which we will call from now on a spectrogram. At the same time, having the spectrum of the observed star (it is a known star), effects such as the atmosphere or the point spread function (cf. section 4) can be simulated to get another spectrogram. Then, the simulated and the observed spectrograms are compared and the simulation is iterated to infer the input parameters describing the atmosphere in the most precise way possible. This is called a forward modelling as the goal is to match the spectrogram by modelling one, unlike backward modelling which consists in extracting the spectra from the observed spectrogram.

3 The dispersion method

3.1 Dispersion with slitless spectroscopy

To get a spectrogram from a source, the light received from it needs to be dispersed. The technology chosen for the Auxtel to disperse the sky is called slitless spectroscopy. Unlike spectroscopy with a slit, the whole field of view is dispersed as shown on figure 5. When doing spectroscopy, there are several solutions. The first one is to use a slit. When using this, the slit needs to be placed precisely on the object to disperse and this might whatsoever cut some of the flux from the source. Another solution is to use an integral field unit (IFU). This splits the field of view into slices which are then dispersed. This produces a three-dimensional cube containing two dimensions describing the spatial variations and one containing the wavelengths. This is a very efficient technique which is also very expensive. This is why in Auxtel, the technique used is called slitless spectroscopy. It is much easier to use than spectroscopy with a slit as there is no need to place a slit, it directly disperses the whole field of view. It is also much cheaper than the IFU. However, it is a complex method as there may be cross-contamination between several sources. Indeed, by dispersing them, they may overlap at it is shown in figure 5. Finally, there may also be self-contamination within an extended source.

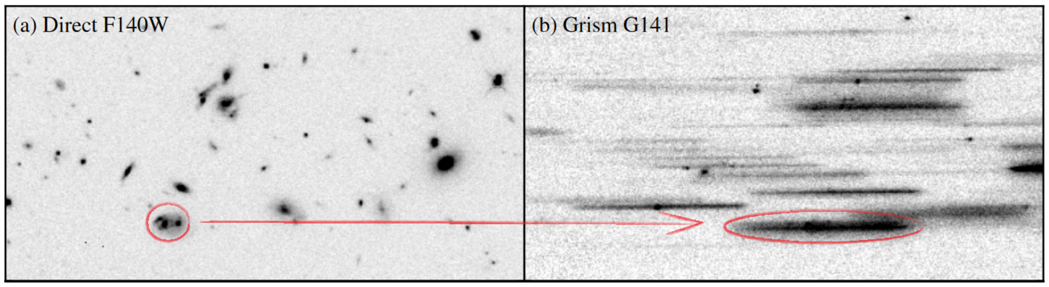
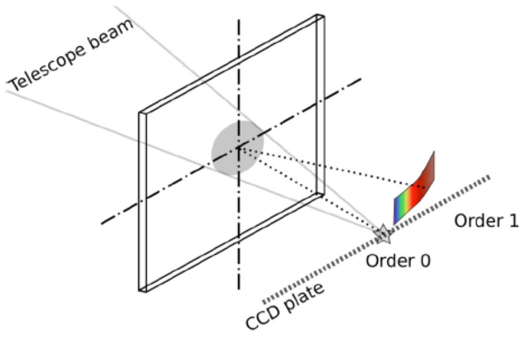


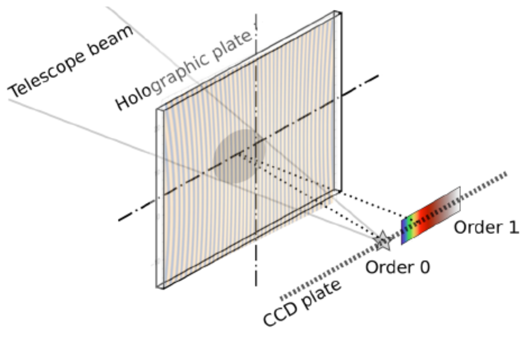
Figure 5: Example of slitless spectroscopy using a grism (grating + prism). The part on the left circled in red becomes the part on the right when dispersed. Adapted from [2].

3.2 The Ronchi grating

Furthermore, the tool used to disperse is a Ronchi grating. The Ronchi grating is a basic diffraction grating consisting in parallel slits on a surface. Its dispersion is characterised by a dispersion law $\Delta(\lambda)$. It is measured with a laboratory test and it is known. A consequent problem is that the Ronchi grating does not project the spectrum on a bi-dimensional plane but on three dimensions as shown on image 6a. This is due to an optical path variation caused by the diffraction angle. Consequently, all wavelength cannot be in focus at the same time and this induces a defocus depending on the wavelength of the dispersed light-ray. A potential solution to this problem which is discussed is the implementation of a Hologram grating, shown in figure 6b.



(a) Sketch of the Ronchi grating showing how the images are defocused depending on their wavelength



(b) Sketch of the Hologram grating showing how the images are now all in focus on the same bi-dimensional plane.

Figure 6: Gratings considered for the Auxtel. The images are from a LSST conference.

The hologram grating is like a normal grating but forces focus on the focal plane at all wavelength. Hence, the 0^{th} and the 1^{st} order both focus wholly on the same plane. Another option which was not retained as it was far more expensive is the use of a grism (a grating plus a prism). With this tool, higher orders could be mainly removed. The second order could become 1% of the first order in intensity so that cross-contamination could be diminished. Also, the first order of the dispersion -the spectrum- could have been accentuated this way.

4 Point spread function (PSF)

4.1 Principle

When imaging, an important part is called the point spread function. The PSF is the function returned by the imaging system when observing a punctual source (typically a star in astrophysics) which will convolve the true image as shown in figure 7.

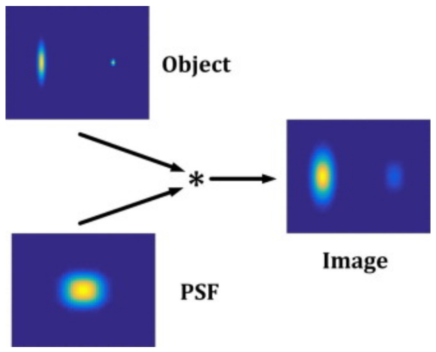


Figure 7: Example of the effect of the PSF on the image observed. Here, the object (upper left) is convolved by the PSF (bottom left) which results in the image observed (right). Image taken from [3]

The imaging system is not perfect and will alter the image observed. Even if it was perfect, due to its size it would induce diffraction and this will limit the resolution of the telescope. In addition to that, for ground-based telescopes, the atmosphere will also blur the impulse response, this is called the seeing. When there is an atmosphere (i.e. for non-spatial telescopes), the seeing overcomes the diffraction and is the main component (cf section 5). One of the methods used to reduce the impact of the atmosphere in real time is called adaptative optics. It is a strenuous and expensive task so an alternative was used which was to model the PSF. Then, by matching the image convolved by the modelled PSF to the observed spectrum, the effect of the PSF is modelled and the atmospheric parameters are inferred. The more precise the PSF is, the better the result will be. Hence the need to have a very precise characterisation of it.

4.2 Full width half maximum (FWHM)

A simplified quantity exhibiting the angular resolution limit of a telescope is the full width half-maximum (FWHM) of the PSF. When only considering diffraction by a circular aperture, the PSF is an Airy disc (a Bessel function) and its FWHM is given by $\sim \frac{\lambda}{D}$. However, it is seldom the case as another parameter has to be taken into consideration, Fried's parameter r_0. This is a common parameter used to describe turbulence in the atmosphere (detailed in section 5). The limit between the diffraction-limited case and the seeing-limited case is set by the relation of Fried's parameter to the size of the dish of the telescope. If $r_0 < D$, the image is limited by the atmosphere whereas in the opposite case, it is diffraction limited. The *seeing* is of significant interest to distinguish *good* and *bad* nights. In most modern telescopes $D \gg r_0$ and to reach the diffraction limit, adaptive optics must be implemented.

5 Atmospheric turbulence

5.1 Kolmogorov model

Kolmogorov made the hypothesis that at intermediate scales, turbulences were homogeneous and isotropic. He then derived quantities describing the properties statistically of the turbulence, such as their power spectrum. It was further developed by Tatarskiĭ [4] who described the index variation of the atmosphere as a phase variation. Assuming a random Gaussian distribution, the phase structure function is described by equation (1) [5], $\Gamma(x)$ being the gamma function and r_0 Fried's parameter.

$$D_\Phi(\mathbf{r}) = 2\left[\frac{24}{5}\Gamma\left(\frac{6}{5}\right)\right]^{5/6}\left(\frac{\mathbf{r}}{r_0}\right)^{\frac{5}{3}} \tag{1}$$

r_0 is also called coherence length and is often used to describe turbulence in the atmosphere. It is defined as the length over which the variance of the distortions in space is equal to 1 rad^2. It can be seen more comprehensively as the equivalent diameter of a telescope at which the limitation due to the atmosphere becomes preponderant. r_0 varies as $\lambda^{\frac{6}{5}}$ and decreases as turbulence increases. In the Kolmogorov model, the seeing is given by equation (2) [6].

$$fwhm = 0.976\frac{\lambda}{r_0} \tag{2}$$

The structure function is related to the optical transfer function of the telescope as shown in equation (3). Then the atmospheric PSF if given by the Fourier transform of $T_0(f)$.

$$T_o(\mathbf{f}) = \exp\left(-\frac{1}{2}D_\phi(\lambda\mathbf{f})\right) \tag{3}$$

5.2 Extension to Kolmogorov's model

An extent to the Kolmogorov model is the Von Karman model which includes another parameter, the outer scale L_0. It is the larger scale of the perturbations and describes a *layer* of the atmosphere. Then, the new phase variation is given by equation (4) [7] where $K_{\frac{5}{6}}$ is the modified Bessel function of the third kind (or McDonald function).

$$D_\Phi(r) = \frac{\Gamma\left(\frac{11}{6}\right)}{2^{11/6}\pi^{8/3}}\left[\frac{24}{5}\Gamma\left(\frac{6}{5}\right)\right]^{5/6}\left(\frac{r_0}{L_0}\right)^{-5/3}\left[2^{-1/6}\Gamma\left(\frac{5}{6}\right) - \left(\frac{2\pi r}{L_0}\right)^{5/6} K_{5/6}\left(\frac{2\pi r}{L_0}\right)\right] \tag{4}$$

Also, in this model, the fwhm is given by equation (5).

$$fwhm = \frac{0.976\lambda}{r_0}.\sqrt{1 - 2.183\left(\frac{r_0}{L_0}\right)^{0.356}} \tag{5}$$

There are also others parameters which may be taken into account such as the wind-speed v_0. If there is almost no wind, this can be neglected for short poses whereas, for high winds, some images may become unusable. From it, a quantity similar to r_0 can be determined which is the characteristic time of variation of the turbulence τ_0, with $\tau_0 \approx 0.314\frac{r_0}{v_0}$.

6 Fourier optics

Light has a wave-particle duality and there are two main ways to study its propagation: geometrical optics and Fourier optics. However, if geometrical optics are an intuitive approach to the propagation of light, a limit is met when interferences are considered. If the diffraction phenomenon is not negligible, a more fundamental approach such as Fourier optics is needed. In Fourier optics, the light is described as a wave and is based on Maxwell's equations of electromagnetism and uses the Fourier transform described by equation (6) for a function f.

$$\hat{f}(\vec{k}) = \iint_{-\infty}^{\infty} f(\vec{r}).e^{-i\vec{k}\vec{r}}d\vec{r} \tag{6}$$

7 Mathematical description of the image observed

A source observed is characterised by its intrinsic flux distribution $C(\vec{r},\lambda)$. This is the quantity wanted when looking at a source. Using P_0 to denote the impulse response function (the instrumental PSF and the seeing, section 4) and $\vec{\Delta}$ for the dispersion law of the grating (section 3.2), the observed image $I(\vec{r})$ is given by (7). The convolution is represented by an asterisk $*$.

$$I(\vec{r}) = \int (C * P_0)(\vec{r} - \vec{\Delta}(\lambda), \lambda) d\lambda \tag{7}$$

By using a Fourier transform, the equation becomes equation (8).

$$\hat{I}(\vec{k}) = \int \hat{C}(\vec{k},\lambda)\hat{P}_0(\vec{k},\lambda) e^{-2i\pi\vec{k}\vec{\Delta}(\lambda)} d\lambda \tag{8}$$

If the source is considered separable, it can be described by $C(\vec{r},\lambda) = F(\vec{r}) \times S(\lambda)$, S being its spectrum. If it is furthermore punctual, as a star, $F(r) = \delta(r)$ which would lead to equation (9) by taking its inverse Fourier transform.

$$I(\vec{r}) \approx FT^{-1}\left(\int \hat{P}_0(\vec{k},\lambda) S(\lambda) e^{-2i\pi\vec{k}\vec{\Delta}(\lambda)} d\lambda\right) \tag{9}$$

Then, the parameters left are the PSF $P_0(\vec{k},\lambda)$, the spectrum of the source $S(\lambda)$ and the dispersion law $\Delta(\lambda)$ which consequently simplifies the model. The dispersion law is composed of the dispersion from the grating which is known may contain some other effect due to the optics. The observed star is known so its spectrum is referenced.
Thus, the main thing to model is the PSF $P_0(\vec{k},\lambda)$ for which the major components are the defocus due to the Ronchi grating and the atmospheric turbulence. Also, an effect called the atmospheric differential refraction was included in the dispersion law $\Delta(\lambda)$ as it shifts the spectrum and is a similar effect. This is the work which will be presented here.

I will now introduce how the basis for these models was built and then detail the 3 models mentioned earlier; respectively the defocus, the atmospheric turbulence and the atmospheric differential refraction. Finally, I am going to conclude on the accuracy of these models and how they could be improved.

Part II
First steps of the model

1 Tools for the models

At first, I was advised to use python libraries so I studied several of them. Some were advised to me but I found several others. I compared all these libraries in detail, checked if they were regularly updated and checked what their main purpose was. In table 1 is shown some of the parameters that were seen for several libraries.

Library	Poppy [8, 9]	OpticSpy [10]	Soapy [11]	HCIPy [12]
Optical simulation	✓	✓	✓	✓
Defocus	✓			✓
Atmosphere			✓	✓
Adaptive optics			✓	✓

Table 1: Comparison of the python libraries

This is a non-exhaustive list as other libraries such as WebbPSF [13] and others were also investigated but deemed less adapted or relevant for this study. The green ticks represent the key tools needed for this work. There is a function *ThinLens* in HCIPy which supposedly models the defocus. After trying to use it and looking at the source code, I found out that it was not implemented yet and thus did nothing. In HCIPy, the modelling of the defocus was not a built-in function and I had to implement it myself, hence the orange tick in the table. The most interesting libraries for this study were Poppy, OpticSpy, Soapy and HCIPy as they all allowed the simulation of optics. Opticspy contained several errors. I corrected some but it still did not seem reliable so

I deemed reasonable not to use it. HCIPy is a library for High Contrast Imaging which includes adaptive optics simulation, coronagraphy and optical diffraction calculations, more precisely both Fresnel and Fraunhofer approximations. I found some minor mistakes but it was globally thorough. POPPy is a simulation package of optical propagation developed for the James Webb Space Telescope (JWST) which logically did not include the atmosphere as it is for a space telescope. However, it included the implementation of a lens to defocus the PSF. Finally, Soapy is a Montecarlo adaptive optics simulation tool and consequently includes the possibility to model the atmosphere. It must also be noted that I used Astropy [14, 15], Numpy [16], Scipy [17], Matplotlib [18] and iPython [19] in this work for plotting, mathematical or physics functions and several tests.

2 Optical propagation

2.1 Propagation to the focal plane

As shown in figure 8, two planes are considered.

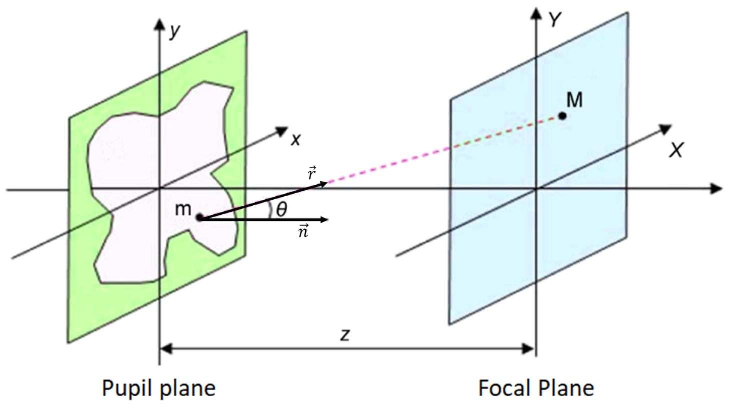


Figure 8: Sketch of the pupil and the focal plane. Adapted from [20]

The first one is the pupil plane -or aperture plane- (0,x,y). This is where the rays arrive from stars at infinity (i.e. the primary mirror in our case). The second one is the focal plane -or image plane- (0',X,Y), where the image is observed, and placed at a distance z from the aperture plane. This corresponds to the CCD plane in the Auxtel. The amplitude of the incident wave is noted U(x,y) in the pupil plane and becomes V(X,Y) on the focal plane. If we consider a wave with a wavelength λ, the wave number describing it is $k = 2\pi/\lambda = 2\pi\nu$ with ν being the frequency of the wave. Then, when considering a point m(x,y) situated on the pupil plane, its image on the focal plane is described by equation (10) with $H(x, y, X, Y)$ given in equation (11) [21].

$$V(X,Y) = \iint_{-\infty}^{\infty} H(x,y,X,Y)U(x,y)dxdy \tag{10}$$

$$H(x,y,X,Y) = \frac{1}{i\lambda z}\frac{\exp(i\vec{k}\vec{r})}{r}\cos(\vec{n},\vec{r}) \tag{11}$$

$\vec{n}$ and $\vec{r}$ are respectively the normal to the pupil plane and the vector from m(x,y) to M(X,Y). Naturally, $U(x,y) = 0$ outside of the aperture.

2.2 Fresnel and Fraunhofer diffraction

If we consider an aperture of size D at a distance d from the focal plane, the phase difference between the centre and the edge of the image on each point of the focal plane is given by $\Delta\Phi = k(d_{edge} - d)$, with k being the wavenumber. By considering that $d_{edge} \approx \sqrt{d^2 + D^2}$, the phase difference is given by equation (12).

$$\Delta\Phi = k\left(\sqrt{d^2 + D^2} - d\right) \approx k\frac{D^2}{2d} \tag{12}$$

Then, two approximation can be made. Either $d \ll kD^2$ (i.e. $\Delta\Phi \gg 1$) or $d \ll kD^2$ (i.e. $\Delta\Phi \ll 1$). These correspond respectively to Fresnel diffraction which is called the near-field approximation and to Fraunhofer diffraction which corresponds to the far field. A rough limit is characterised by the Fresnel number given in equation (13).

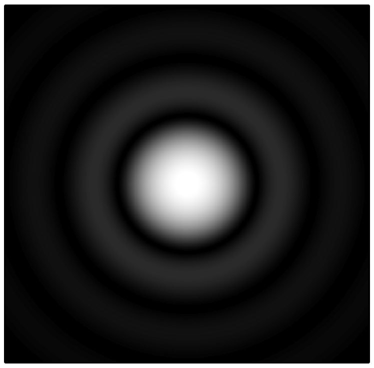
$$F = \frac{D^2}{d\lambda} \tag{13}$$

When $F \gtrsim 1$, the Fresnel diffraction is considered and when $F \ll 1$, the Fraunhofer diffraction is considered. Fresnel diffraction is used for small distances, mainly near pupil planes and close to the z-axis (the axis going from the centre of the focal plane to the centre of the pupil plane). For example, it is used for surface errors on lenses or mirrors, simulating microlenses or scintillation from the atmosphere. On the other hand, Fraunhofer diffraction is used to describe diffraction *far* from the diffracting object. In the ideal conditions, the source and the image are at infinity and the incoming wave is a plane wave [22]. If the focal plane is not at infinity but a converging lens is used, Fraunhofer approximation can be applied [23, 22]. In the present case, the incoming wave is indeed a plane wave as the observed object is a star (considered at infinity) and even though the image is not at infinity, the primary mirror is acting as a converging lens which makes the Fraunhofer diffraction appropriate.

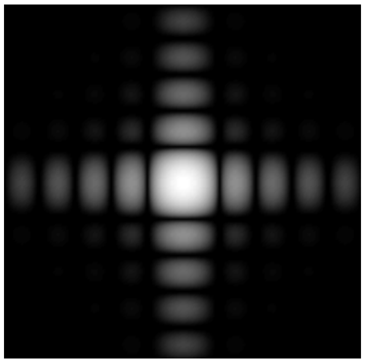
3 Aperture simulation

3.1 Shape of the aperture

With the three libraries used here, the first thing to model was the aperture of the telescope. Indeed, the PSF greatly depends on the shape of the aperture. For the diffraction, a circular PSF would give circumscribing circle as shown in figure 9a whereas a square aperture would give out the shape of a cross as shown in figure 9b.



(a) Diffraction pattern for a circular aperture.



(b) Diffraction pattern for a square aperture.

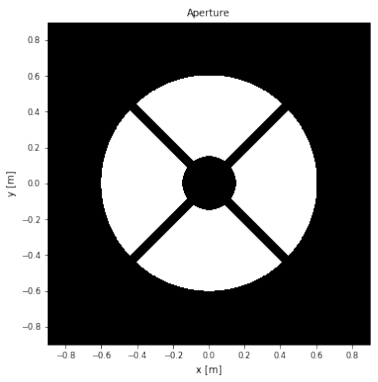
Figure 9: Diffraction pattern for a circular and a square aperture. Both images are from Wikipedia.

For HCIPy and POPPy I made telescope apertures with 2 variables quantities. One defining the obscuration ratio of the aperture and the other the width of the spiders. The diameter of the Auxtel is 1.2m so I used this one by default. Soapy parameters need to be included as a .yaml file and adds an obscuration ratio but no spiders. A mask to model the spiders could be used but as they should not induce a great difference, I just used a circular aperture with obscuration.

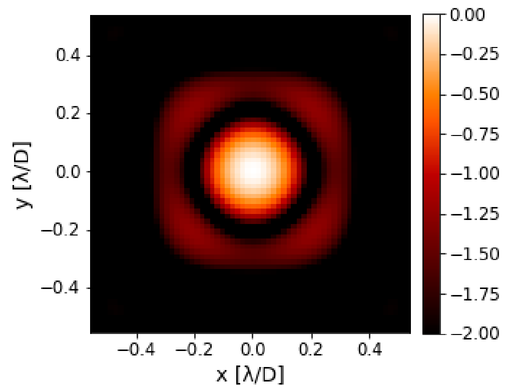
3.2 HCIPy

With the library HCIPy I modelled the Auxtel with variable obscuration ratio and spider width as shown in figure 10a. In HCIPy, no lenses were implemented to model the convergence of the telescope. Fraunhofer was thus used to model an image very far or with a modified wavelength. Indeed, by modifying the wavelength, the value of the F-number can be compensated to reach the wanted approximation. As the choice was to use either Fresnel diffraction which was made to be applied near the pupil plane or Fraunhofer diffraction with tweaked parameters, it was not obvious which regime to choose. The documentation in the library used unrealistic values ($\lambda = 1$) to model diffraction in order to have a Fresnel number $F \ll 1$ with a telescope of 6.5m so that the Fraunhofer diffraction approximation would be correct. However, with the parameters used, $F \sim 42$ and yet the Fraunhofer diffraction was applied. This nonetheless gave a coherent result. As with a wavelength of $\lambda = 1000$ nm and a focal length of 21.6m (the f-number is 18), the Fresnel number is $F \sim 6.66 \times 10^4 \gg 1$, I first tried to use

Fresnel diffraction. However it did not offer the results I expected, even by tweaking the input parameters. By contacting the developer I was told that indeed, in this case and more generally to model diffraction in telescopes, Fraunhofer diffraction was used. Indeed, the telescope consists in observing incoming parallel plane-waves with either a converging lens or focusing at infinity. Consequently, I used Fraunhofer diffraction and tweaked the focal length to reach $F \sim 6.66$. This was done by using a focal length of $f_1 = 10^4 f$ with f being the real focal length of the Auxtel. This gave the result shown in figure 10b which is the diffraction pattern due to to the telescope. This is the expected diffraction pattern (i.e. a central disc surrounded by a fainter disc). However, the units which were computed and automatically plotted on this graph do not seem coherent as $1,22\lambda/D$ is supposed to be the first minimum (theoretically 0) of the Airy disc. Here, the scale displays that it is at $\sim 0,2\lambda/D$. λ/D is an angle in radians and can be converted in arcsecond. $1rad \sim \frac{180*3600}{\pi} arcsec$. This gives that $1\frac{\lambda}{D} \sim 0.17arcsec$ so I suspect a mistake was made and that the units are in reality arcsec.



(a) Image of the model used with HCIPy to simulate the aperture of the telescope.

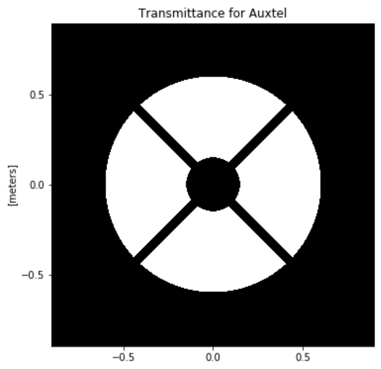


(b) Image of the point spread function of the telescope shown in figure 10a with a logarithmic scale.

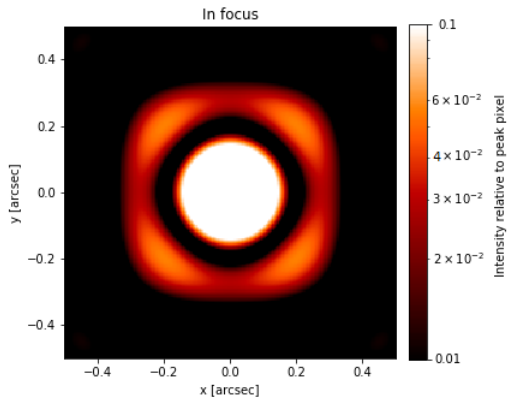
Figure 10: Modelling of the PSF of the aperture of Auxtel with HCIPy.

3.3 POPPy

As with HCIPy, I modelled a variable obscuration ratio and spider width with POPPy to create an aperture similar to the the one of Auxtel as shown in figure 11a. This was more straightforward as it is a library designed for similar telescopes. The diffraction pattern obtained is shown in figure 11b.



(a) Image of the model used with HCIPy to simulate the aperture of the telescope.



(b) Image of the point spread function of the telescope shown in figure 11a.

Figure 11: Modelling of the PSF of the aperture of Auxtel with POPPy.

The image obtained is an Airy disc as expected. Furthermore, it looks a lot like the result from HCPIy and has the same scales with a unit in arcsec and not in λ/D. This did tend to confirm that there is an error in the units of HCIPy.

4 Comparison of the diffraction PSF from HCIPy and POPPy

All the following tests have been done with the same parameters, i.e. $\lambda = 1000nm$, 4 spiders of 0.05m and the parameters of Auxtel. A first thing to notice on the plots done is their scale, as mentioned previously. When POPPy outputs a scale in arcsecond, HCIPy outputs them in λ/D. As a reminder, $1\frac{\lambda}{D} \sim 0.17arcsec$. However, by comparing both results shown in figure 12, even though they match quite well, there is a small difference. To match them, the x-axis of HCIPy must be multiplied by 0.95, as shown in figure 13. I tried to find out what the source of this discrepancy was but could not find it. I suppose that it is a subtle effect in HCIPy or a problem linked to the way of defining it. It is likely that the arcsecond of POPPy is exact (furthermore it is a more developed library). For the next plots, I re-scaled them to be able to compare them. When the PSF is re-scaled, both the sums and the slices match quite well with a discrepancy of 10^{-2} for the slice and 10^{-1} for the sum.

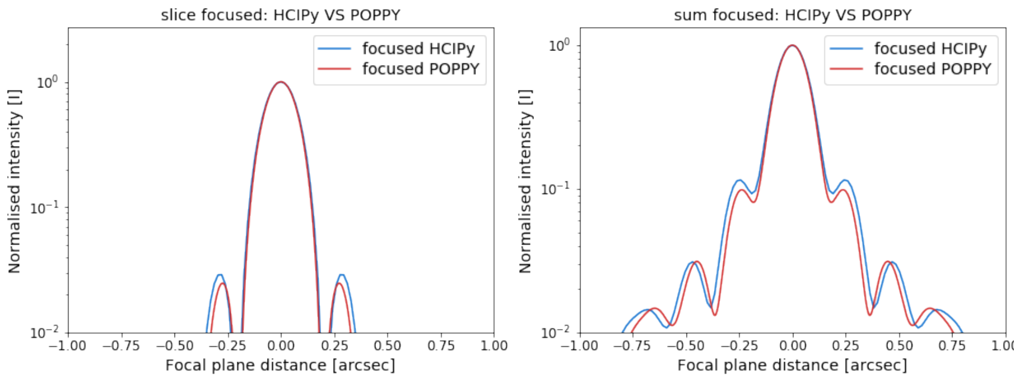


Figure 12: Plot of the sum and the slice of the PSF without re-scaling.

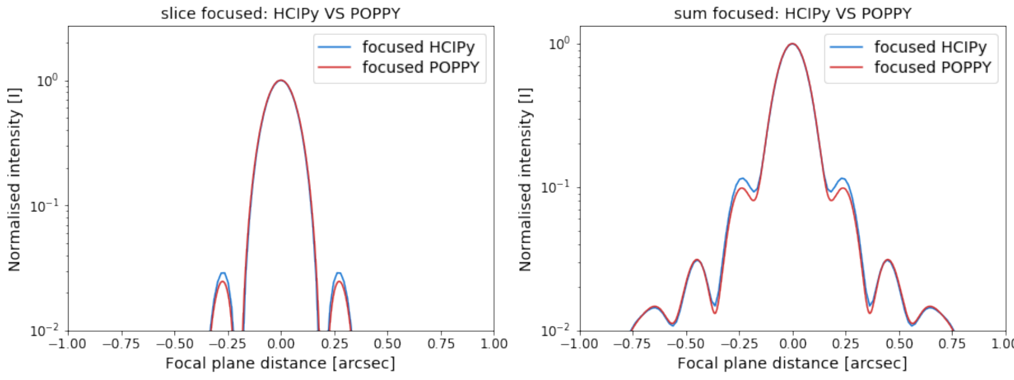


Figure 13: Plot of the sum and the slice of the PSF re-scaled by a factor 1.9.

Also, these graphs (and all the others simulated afterwards) were done simulating a high sampling of the CCD screen. However, if this value is lowered, for example having a pixel size of 0.1 microns instead of 10 microns, it limits the precision as shown in figure 14.

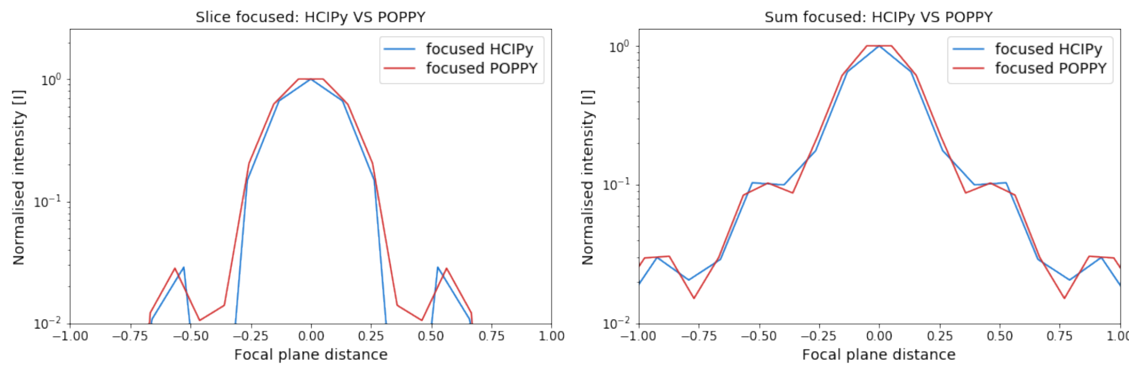


Figure 14: Plot of the sum and the slice of the PSF with a low sampling.

Part III
Defocus

As explained in 3.2, the Ronchi grating induced a defocus depending on the wavelength considered. This is the first thing I tried to model since it was expected to be the major source of error.

1 Theory to model the defocus

To model the defocus, there were two possible approaches: either implementing a lens which would make the image in focus a bit before the focal plane (section 1.1) or including the defocus term through Zernike polynomials (section 1.2). As previously said, in HCIPy there is a function *ThinLens* which supposedly models the defocus. It was not implemented yet so I tried to implement both the lens and the Zernike polynomials methods by myself. POPPy included a lens so this was more straightforward.

1.1 Lens implementation

In Fourier optics, the light can be described by equation (14), A being its amplitude and ϕ its phase.

$$\psi(\vec{r}) = A(\vec{r})e^{i\phi(\vec{r})} \tag{14}$$

The lens is described by the analytic function given by equation (15) in Fourier space.

$$t(x,y) = Be^{i\pi\frac{x^2+y^2}{\lambda f}} \tag{15}$$

As this is used on a normalised beam and the amplitude is irrelevant, to model a lens, it is sufficient to multiply the wavefront by the phase shift $e^{i\pi\frac{x^2+y^2}{\lambda f}}$. This is equivalent to applying a phase mask to the wave.

1.2 Defocus with Zernike polynomials

To use the Zernike polynomials for the defocus, two parts have to be considered. The Zernike polynomial itself (section 1.2.1) and the sagitta difference (section 1.2.2). How to use them both to model the defocus is detailed in section 1.2.3.

1.2.1 Third Zernike polynomial

The first part to consider is Zernike polynomials. As they are an orthogonal basis on the unit disc, they are used to describe aberrations in optical systems. The third Zernike polynomial Z_2^0 describes the defocus. In polar coordinates (ρ, φ), even Zernike polynomials are described by equation (16) and equation (17).

$$Z_n^m(\rho,\varphi) = R_n^m(\rho)cos(m\varphi) \tag{16}$$

$$R_2^0(\rho) = \sum_{k=0}^{\frac{n-m}{2}} \frac{(-1)^k\,(n-k)!}{k!\,\left(\frac{n+m}{2}-k\right)!\,\left(\frac{n-m}{2}-k\right)!}\,\rho^{n-2\,k} \tag{17}$$

In the case $(m,n) = (0,2)$, it can be simplified and is described by (18)[24].

$$Z_2^0(\rho,\varphi) = R_2^0(\rho) = \sqrt{3}(2\rho^2 - 1) \tag{18}$$

There is no angular component as the effect is a perfect longitudinal defocus and is symmetric by rotation. Furthermore, there are variations in the equation depending on the shape of the aperture as detailed in [25, 26]. For an annular aperture with an obscuration ratio $\epsilon = \frac{radius_{aperture}}{radius_{obstruction}}$, computations are a bit more complex and the third Zernike polynomial is given by equation (19) [27].

$$Z_2^0(\rho,\varphi,\epsilon) = R_2^0(\rho,\epsilon) = \frac{2\rho^2 - 1 - \epsilon^2}{1-\epsilon^2} \tag{19}$$

This is the kind of equations used in the source code to describe the defocus. I did not use them directly but they are of use here to understand the overall idea.

1.2.2 Sagitta difference

The other part to consider is the peak-to-valley (P-V) error. This error, when due to the defocus, corresponds to the difference between the sagitta of two spheres whose centre is respectively situated at the focal point and the point of observation. This is the quantity noted W in figure 15.

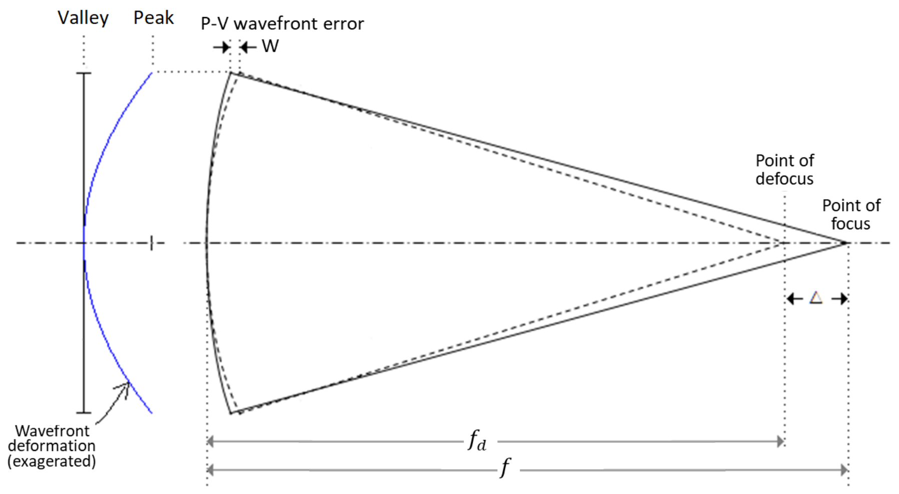


Figure 15: Sketch showing the effect of sagitta difference on the peak-to-valley error. Adapted from [28]

This P-V error is given by the expansion series shown in equation (20) [28, 29, 30] where Δ is the distance between the two centres, D is the diameter of the pupil (i.e. of the telescope) and f is the focal length.

$$W = \left(\frac{1}{f-\Delta} - \frac{1}{f}\right)\frac{D^2}{8} + \left(\frac{1}{(f-\Delta)^3} - \frac{1}{f^3}\right)\frac{D^4}{32} + \mathcal{O}\left(\left(\frac{D}{f}\right)^3\right) \tag{20}$$

If $\Delta \ll D$, this expression can be further simplified into equation (21) with $F = f/D$ being the f-number.

$$W \approx \frac{\Delta D^2}{8f^2} = \frac{\Delta}{8F^2} \tag{21}$$

By multiplying the P-V error W by the wave-number $k = \frac{2\pi}{\lambda}$, the corresponding phase shift can be computed. This phase shift is then given by equation (22). It corresponds to the peak-to-valley difference in phase due to the defocus.

$$\Phi = \frac{\Delta}{8F^2}\frac{2\pi}{\lambda} \tag{22}$$

1.2.3 Application

To include the effect of the defocus, two elements are needed. The first one is the phase shift corresponding to the optical path difference due to the defocus. The second one is the values of the third Zernike polynomial. By then normalising the peak-to-valley value of the third Zernike polynomial with the computed phase shift, the defocus was implemented [28, 30]. However, it is important to note that the approximation $\Delta \ll D$ was made for equation (21) and that this means that this computation is only viable for a small defocus.

2 Model of a lens with HCIPy

2.1 Consistency check

At first, I tried to model a lens in Fourier space as introduced in section 1.1. The first consistency check was to retrieve the image of the instrumental PSF with no defocus. To do this, the focal f_2 used for the lens just had to be much greater than the focal length of the telescope as it should not change the diffracted image obtained. It was indeed the case. However, something to take into account was that the distance to the focal plane was multiplied by 10^4 to have a coherent Fraunhofer propagation. So the distance to the screen considered was $f_1 = 10^4 f$ and not just f. To have a coherent result, this had to be noted when considering the focal f_2 to use for the lens in equation (15). To retrieve the focused image with a difference in intensity being less than 10^{-3}, f_2 (the focal length of the lens) needed to be at least $10^2 \times 10^4 f$. For an additional schematic presentation of the notation, refer to figure 16.

2.2 Constraints for defocus

The next step was to add the defocus term. To be able to see a difference in the image while still having an image looking like a PSF, f_2 needed to be of the order of $10^3 f$ to $10^5 f$. If it was bigger, the changes would not be seen and if it was lower, it would not look like the PSF anymore.

2.3 Adding the defocus

The way to implement the defocus was not trivial. Hence, I tried several methods shown in figure 16.

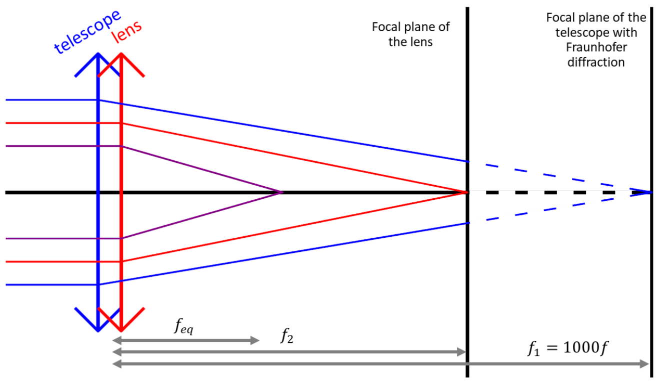


Figure 16: The telescope and the converging lens have been separated for sketching purposes but there is no space between them. In blue is the trajectory of the light-rays if the additional lens (red) is not taken into account and considering that the telescope does act as a lens. In red is the same trajectory considering that the telescope does not act as a converging lens. In purple is shown the combination of both acting as a lens. The dashed line intend to show that there is actually a great distance between the two planes.

– A first idea may be to use f as f_2. Indeed, as Fraunhofer diffraction is considered, the image is at infinity. Since the waves are supposed to be planar, if the telescope does not act as a converging lens, it should be possible to make them converge at any focal distance. This case would correspond to the red light-rays on figure 16 where the telescope does not act as a lens. However, it did not work as the results were unrealistic.

- Also, it may be thought that the solution was to take $f_2 = 10^4(f - \Delta)$. Indeed, if the waves are parallel, we can make them focus a bit behind the focal screen. This would correspond to a blue-like scenario in figure 16 where the lens makes the rays converge. However, it did not work which may mean that the code induces an effect similar to a converging lens with the telescope. Then, adding another one made it converge even more. Nonetheless, by changing the position of the screen, I did not manage to retrieve anything realistic.
- By supposing that the Fraunhofer diffraction was focusing at infinity and as the lens equivalent to two lenses is given by equation (23), I used this to know how to input the defocus.

$$\frac{1}{f_{eq}} = \frac{1}{f_1} + \frac{1}{f_2} \tag{23}$$

  By considering that I wanted $f_{eq} = f - \Delta$ (delta being the defocus and f the focal length of the telescope) and that $f_1 = 10^4 f$ was the default convergence of the Fraunhofer diffraction, the value of f_2 used in equation (15) is given by (24).

$$f_2 = \frac{f(f - \Delta)}{f - 10^{-4}(f - \Delta)} \underset{\Delta \ll 10^4}{\approx} f - \Delta \tag{24}$$

  This would correspond to the purple scenario in 16. However, what is not shown on the sketch is that as $f \ll f_1$, the red and the purple paths were almost the same so this was actually like the first case. This is of the order of f and, as expected, this also proved inconclusive.

For all these solutions I tried to change f_1 and the resolution (as I reached resolution limits for some tests) to see if this was a sampling effect but I did not manage to find proper results. Even though the main developer said that *extreme care [must be taken] when sampling* when using a lens with this library, it did not seem like this was the cause for the inability to model the defocus. A better understanding on how the Fraunhofer propagation is modelled in the source code is needed to be able to add the defocus by using a lens. As said in section 2.2, the distance from the pupil plane to the focal plane considered had to be significantly bigger than the real one. Thus, the way to add a longitudinal defocus was not obvious with this approximation. Consequently, I tried to modify the wavefront mathematically with Zernike polynomials.

2.4 Zernike polynomials with HCIPy

Not finding out how to properly implement a lens and this approach being error-prone, I decided to use Zernike polynomials as it was advised by the developer. In HCIPy it was possible to evaluate the third Zernike polynomial on the pupil grid and then extract its peak-to-peak value as explained in section 1.2. We can see the effect of the defocus on the PSF on figure 17.

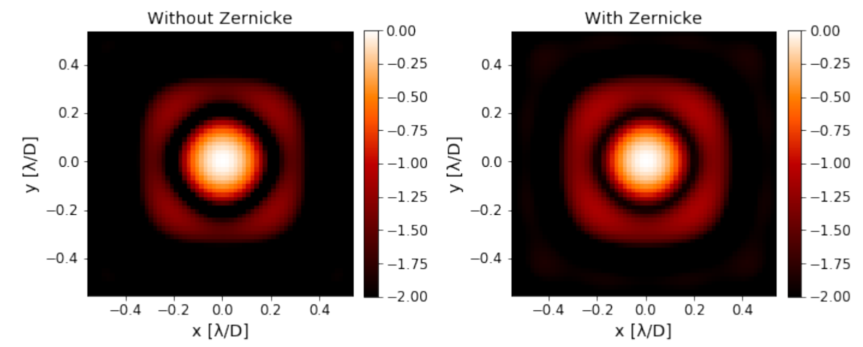
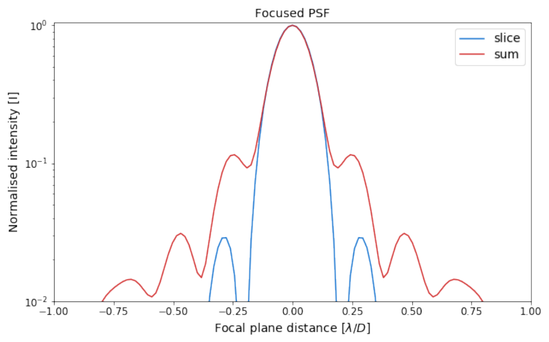
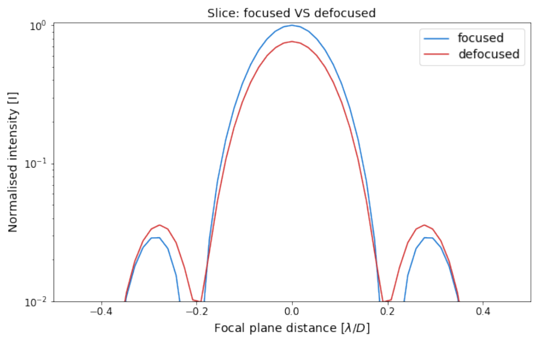


Figure 17: Image of the PSF in focus (left) and with a defocus of 0.001m (right) with a logarithmic scale.

On the left is shown the result without the defocus and on the right with a defocus of 0.001m. Also, to see more precisely the result I also made a plot comparing a slice of the PSF and its sum, shown in figure 18a and a plot comparing the slice of the PSF in focus and the defocused one, shown on figure 18b. On this figure, it can be noted that the intensity decreases in the centre of the PSF and spreads out to the sides.



(a) Plot of the central slice of the sum of the PSF along the y-axis for the focused case (the defocused case is similar).



(b) Slice of the focused and defocused PSF using HCIPy in logarithmic scale. The intensity of the defocused simulation seems to be spread out from the centre.

Figure 18: Comparison of the slice and the sum of the PSF (on the left) and of the focused and defocused PSF (on the right).

3 POPPy

As in this file a function *psf_thinlens* which simulated a lens was already implemented I used it. However, it took a variable called *nwaves* which is the peak-to-valley number of waves of defocus. Having the f-number $F = \frac{f}{D}$ (f being the focal of the telescope and D its diameter) and $k = \frac{2\pi}{\lambda}$, the number of waves of defocus is given by $nwaves = \frac{\Delta k}{8F^2}$. This is what I called the *phase error* in section 1.2. A positive value gives a converging lens and a negative sign a diverging lens which is the opposite sign convention compared to the Zernike polynomials convention. Then, by looking at the source code, I saw that it was, in reality, using Zernike polynomials to add the effect of defocus to the PSF and was consequently similar to the method used in HCIPy. As previously, I did a plot of a slice of the PSF against its sum in 19 and a slice of the PSF in focus against the defocused one in 20b with a defocus of 0.0001m. As with HCIPy, it can be noted the intensity is diminished in the centre and spread out on the borders.

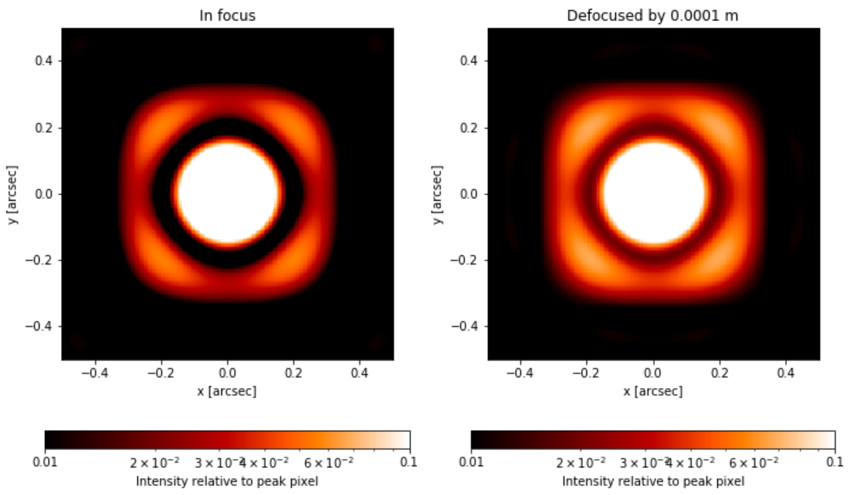
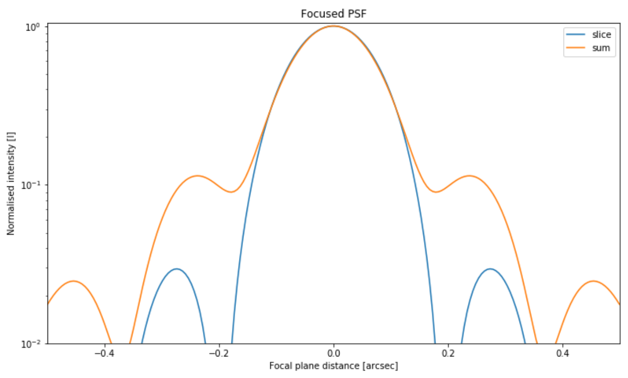
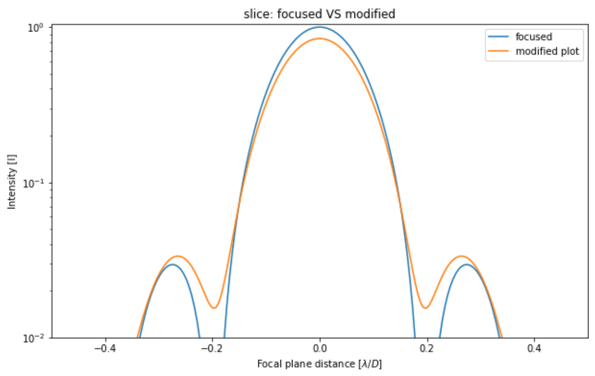


Figure 19: Image of the PSF in focus (left) and with a defocus of 0.0001m (right) with a logarithmic scale.



(a) Plot of the central slice of the sum of the PSF along the y-axis for the focused case (the defocused case is similar).



(b) Slice of the focused and defocused PSF using HCIPy in logarithmic scale. The intensity of the defocused simulation seems to be spread out from the centre.

Figure 20: Comparison of the slice and the sum of the PSF (on the left) and of the focused and defocused PSF (on the right).

4 Comparison between the defocus of HCIPy and POPPy

4.1 Discrepancy

As shown by figure 21 with a defocus of 0.001m, even by taking into account the difference in scale there is a huge difference in the result between the defocused case computed by HCIPy and the defocused case computed by POPPy. By trying to vary the defocus for each of them, I could come close to a match by scaling the shift of POPPy (multiplying it by 0.09). However, this coefficient varied with the defocus I tried to input. For example, with a defocus of 0.003m, it was not possible to match them at all. Since this was not a linear error, it was not an error only due to a difference in the scales but it was an error on the computation method for the defocus.

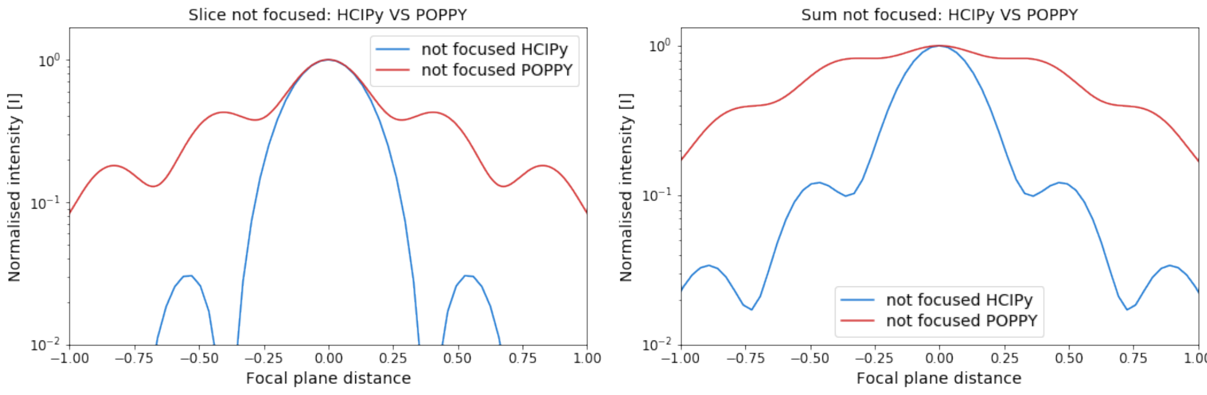


Figure 21: HCIPy and POPPy defocus with the same shift.

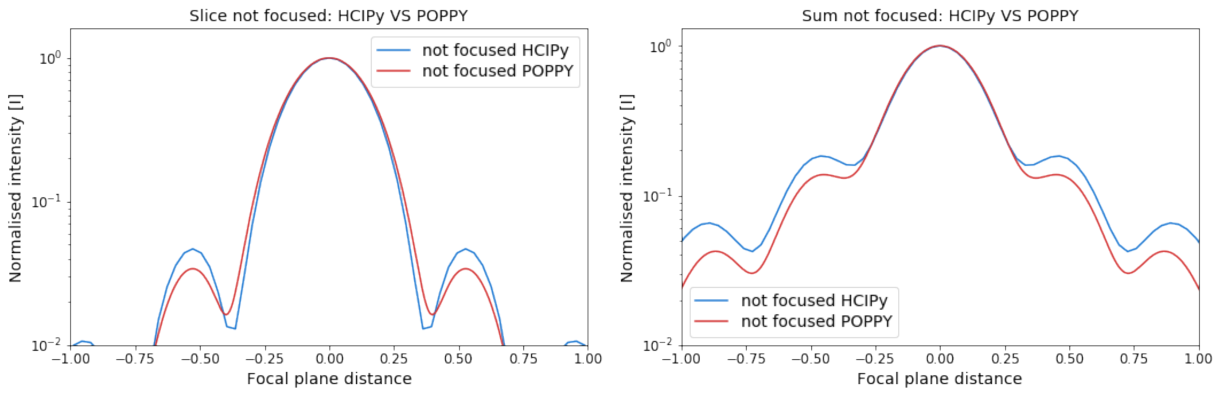


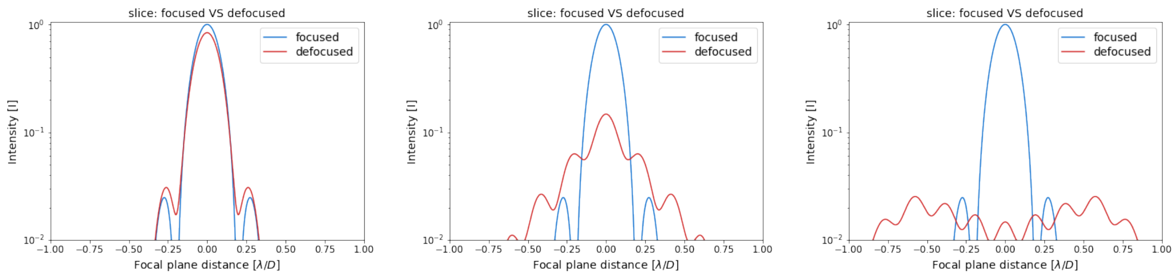
Figure 22: HCIPy and POPPy defocus with the same shift.

4.2 Limits

This discrepancy between both libraries can be shown more clearly as they do not *act* the same way nor on the same ranges with defocus.

4.2.1 POPPy

The defocus shift in POPPy begins to noticeably change the PSF from 0.05mm and then evolves slowly as shown with a defocus of 0.1mm on figure 23a. Afterwards, it considerably changes to reach the figure shown in figure 23b with a defocus of 0.3mm. The intensity of the defocused image then continues to decrease and becomes really faint from 0.5mm (figure 23c).



(a) PSF of the aperture with POPPY defocused by 0.1mm.

(b) PSF of the aperture with POPPY defocused by 0.3mm.

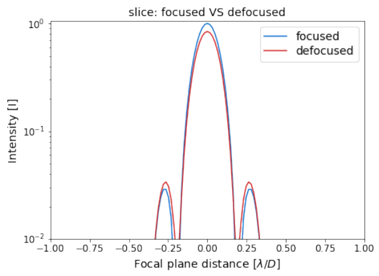
(c) PSF of the aperture with POPPY defocused by 0.5mm.

Figure 23: Comparison of various defocus with POPPY.

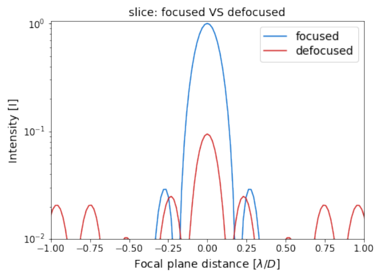
This result seems coherent in a first approach as it varies continuously with all the defocus values tried. However, the effect is very strong. Before 0.1mm, the defocus has almost no influence and after 0.5mm the intensity of the image becomes nothing like the original PSF.

4.2.2 HCIPy

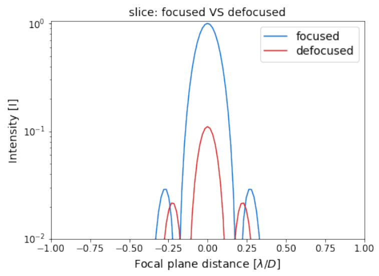
With HCIPy, the shift begins to be noticeable around 0.8 mm (figure 24a) and shifts continuously from there until it reaches 3mm (figure 24a). Then, the intensity in the centre of the PSF increases a bit to reach figure 24c. After reaching 11mm, it doesn't change anymore, at least up to 10m.



(a) PSF of the aperture with HCIPy defocused by 0.8mm.



(b) PSF of the aperture with HCIPy defocused by 3mm.



(c) PSF of the aperture with HCIPy defocused by 11mm.

Figure 24: Comparison of various defocus with HCIPy.

It may look coherent as the intensity of the PSF seems to change continuously but its behaviour changes from a defocus of 3mm and does not change at all from 11mm. This is already surprising but furthermore, these values are quite big for a defocus. I did not expect it to work at these ranges.

4.3 Solution

Seeing these two behaviours, it is obvious that there is a huge difference between those two models and that one (or both) of them is wrong. As I did not know the effect of the defocus on the PSF beforehand, the requirement to attest the adequacy of the model was to input a ”huge” defocus. Indeed, if there is a huge defocus, the image of the aperture should be retrieved. With HCIPy, as said previously, increasing the defocus did not change anything. However, with POPPy, increasing it changed greatly the image as shown in figure 25.

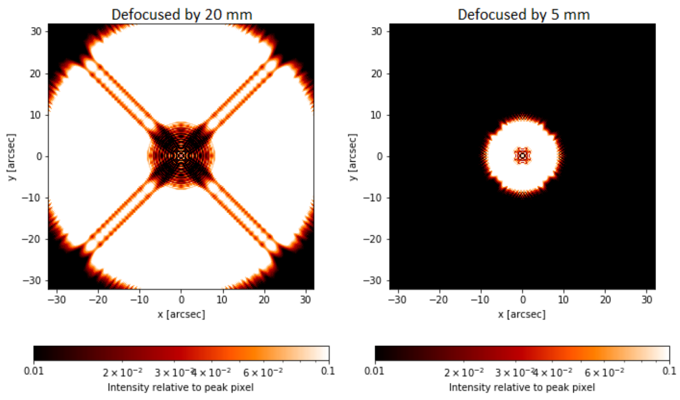


Figure 25: POPPy Instrumental PSF with a defocus of 20 mm (left) and 5mm (right)

The field of view here is not the same as before, it is *zoomed out* compared to the previous plots. It seems that by increasing further the defocus, the images shown here just gets bigger. I tested bigger defocus (with a bigger field of view and a lowered resolution to be able to compute it as it is quite heavy) and this seemed to be confirmed. Nonetheless, I only had the time to compare them at sight, as shown here. I did not plot re-scaled slices of different defocus to see if there were indeed just the same image expanded or if it just seemed that it was the case. As explained earlier, there was an approximation made to use the Zernike polynomials which is that $\Delta \ll D$. Here, that is indeed the case as $20mm \ll 1.2m$.
An alternative solution to using those libraries and Zernike polynomials would be to model a lens as detailed in 1.1. From a 2D intensity picture, by using a 2D Fourier transform on it and multiplying it with equation 15 to re-scale the phase, it may be possible to implement this effect.

Part IV
Atmosphere

1 Atmospheric turbulence (seeing)

1.1 HCIPy

At said in 1, I was first advised to use libraries to have more precision and I used HCIPy. HCIPy includes a function to simulate the atmosphere so I used it at first. It models a single realisation of the atmosphere as a phase screen and accounts for several parameters such as the seeing, the outer scale L_0, the speed of the wind. Then, it propagates the wavefront through this phase screen. It also uses several layers to model the effect of the atmosphere. However, as it produced a single realisation, this had to be averaged because it only gave speckles as shown in figure 26.

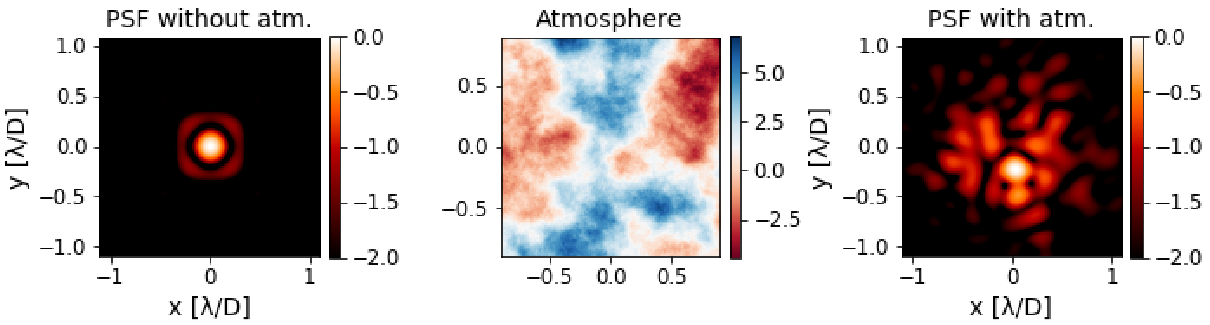


Figure 26: Effect of a realisation of the atmosphere on the PSF. Here is shown the PSF without atmosphere (left), the realisation of the atmosphere (centre) and the modified PSF (right).

Indeed, as the atmosphere is now included, its effect is dominant on the PSF. Since it is an instantaneous single realisation, the image seen is composed of speckles. To have an averaged PSF, I made 100 realisations which gave the result shown in figure 27.

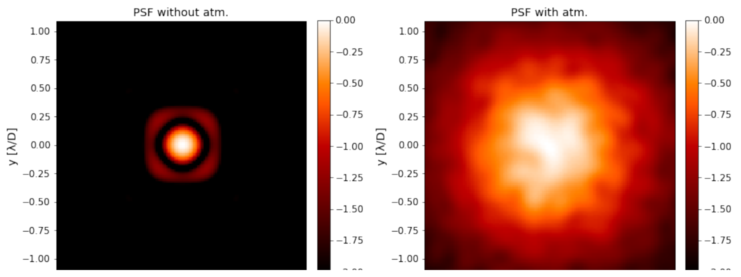


Figure 27: Effect of 100 realisations of the atmosphere averaged on the PSF. Here is shown the PSF without atmosphere (left) and with an average of 100 PSF (right).

Also, in figure 28 is shown as previously the comparison of the slice and the sum for the cases with and without the averaged atmosphere as they differ.

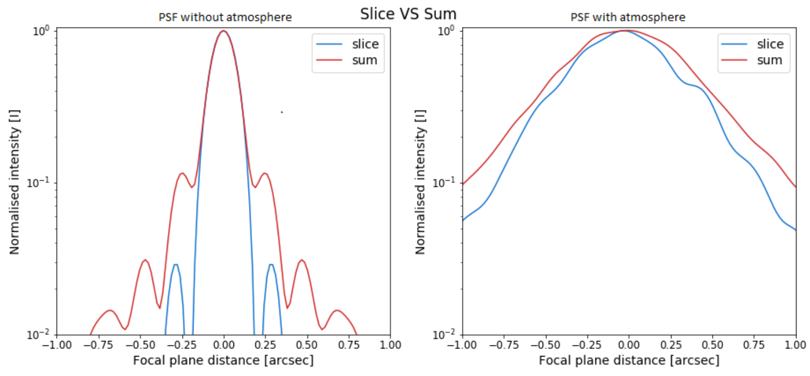


Figure 28: Plot of the central slice and of the sum of the PSF along the y-axis without (left) and with (right) atmosphere.

Finally, in figure 29 is shown the comparison of the slice of the case without atmosphere and the case with atmosphere normalised by the maximum of the case without the atmosphere.

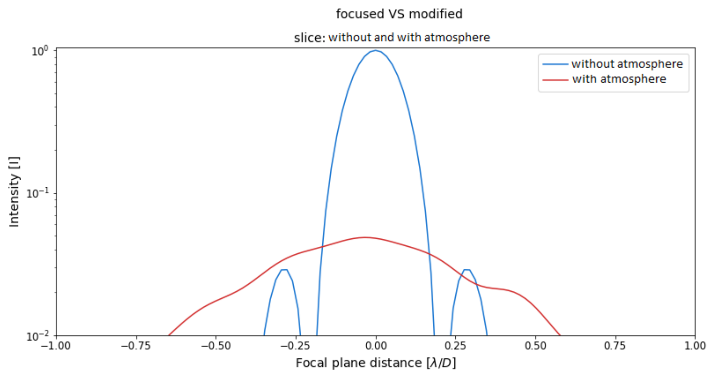


Figure 29: Slice of the PSF without (left) and with (right) atmosphere using HCIPy in logarithmic scale. The intensity of the defocused simulation seems to be spread out from the centre. I plotted here the graph up to 10^{-4} to underline the importance of it on the PSF.

This result seemed reasonable. However, this took in average 15 minutes of computational time and was consequently deemed too long for the simulations.

1.2 Soapy

Another library that I used to model the atmosphere is Soapy which a Montecarlo Adaptive Optic (AO) simulation tool. As adaptative optics aims to correct instantaneously the atmosphere, it also includes a simulation of the atmosphere. I removed all the adaptative optics tools included by default in the simulation to avoid useless computation time and avoid effects not concerning this study. The results were also a single realisation (speckles) and were obtained in approximately 30 seconds. Thus, an average of these realisations was not even done as it would have taken more than 30min to compute. As with the simulation made with HCIPy in section 1.1, it was not needed to make single realisations of the atmosphere. The goal is only to include the average effect of the atmosphere; it is not to compensate its fluctuation in real time with tools used in adaptative optics. It was deemed to slow to be included in the pipeline and was thus not used in the end.

1.3 Using the optical transfer function

1.3.1 Overview

As here the main component of the PSF was the atmosphere, it could be described directly from the Kolmogorov model (section 5). Indeed, the atmospheric PSF can be obtained by a Fourier transform of the optical transfer function. Thus by directly plotting the Fourier transform of (3), it was possible to model this. I adapted a code which was already using these formula [1]. Its use was only to plot the intensity against the radius so I had to adapt it to build a PSF cube. In a first part, I created a 2D grid composed of cell within which was stored the distance from the centre of the grid. From this point on, I iterated this step using an array of wavelengths to produce a cube of PSF.

1.3.2 Comparison of the four computations

In the code used, there were four ways to compute I compared the four of them. Indeed, the Fourier transform of the optical transfer function can be expressed as the Hankel transform of order 0 as shown in equation (25) [1].

$$Hankel_0(T_0(r)) = \int_0^\infty T_0(k) J_0(kr) k dk \tag{25}$$

H_0 is the Hankel transform and J_0 is the first Bessel function of order 0. Thus there are already two ways to compute the Kolmogorov function. Added to that, interpolations of these computations were also implemented to be more computationally efficient. I compared them as shown in figure 30.

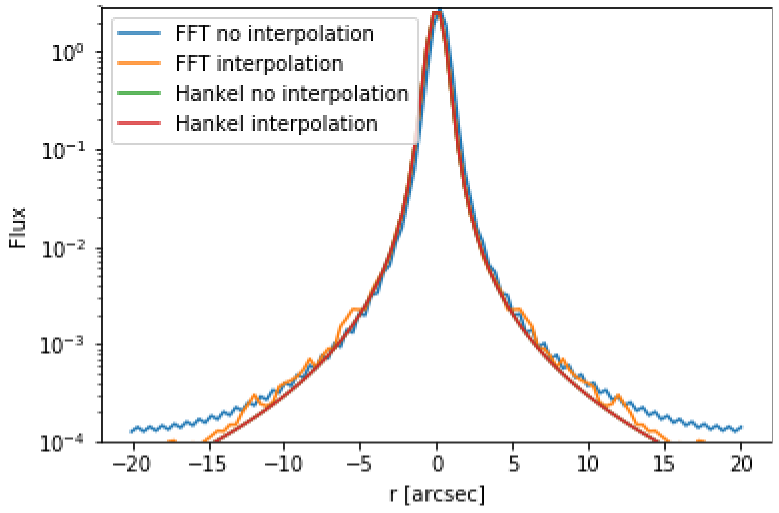


Figure 30: Comparison of the four ways to compute the Kolmogorov model. Using Fourier transform or not and using an interpolation or not.

As the precision wanted for the actual PSF was no further than 10^{-2}, both Hankel transform and the interpolation of the Fourier transform were sufficiently precise. The

1.3.3 Comparison with HCIPy

An interesting comparison to make was to check how this model behaved compared to the result from HCIPy. I have shown the result in figure 31. In blue is the instrumental PSF (from HCIPy without atmosphere) and in red is the results from the simulation of the atmosphere from HCIPy. The other colours correspond to the optical transfer function given in equation 26. I used four different indexes that were compared in [1] to see how they fared with the simulation.

$$\log T_0^i(f) = -\left[\frac{8}{i}\Gamma\left(\frac{2}{i}\right)\right]^{i/2}\left(\frac{\mathbf{r}}{r_0}\right)^i \tag{26}$$

All these curves are normalised by the maximum of the PSF without atmosphere (the blue curve). As the computation time needed for the analytic model was very low (unlike for the HCIPy simulation) and as both results are close, this was a satisfying result. The discrepancy of order 10^{-2} is due to the simulation being a more developed model including the outer scale L_0. By using the Von Karman optical transfer function, a closer match should be reached. Furthermore, as the simulation is an averaged over 100 realisations, it fluctuates a bit. More realisations may lead to a closer match.

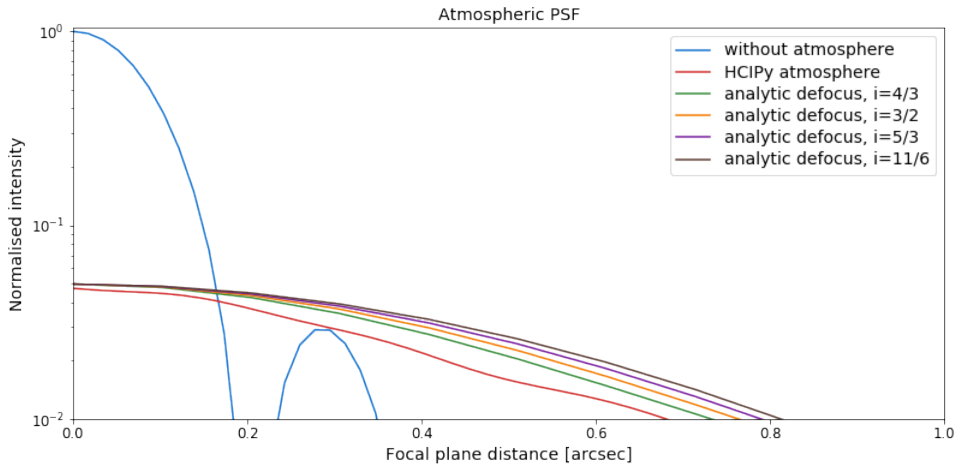


Figure 31: Comparison of the simulation of the atmosphere from HCIPy and from the optical transfer function.

1.3.4 Implementation

Having these four being in reasonable agreement and also having plotted the Kolmogorov PSF as shown in figure 32, it was now trivial to iterate this to get a cube of PSF depending on the wavelength considered. As I ran short on time, however, I did not optimise computationally this code and could not compare the speed of each of the four methods. Also, I did not have the time to implement it in a first iterative code mimicking the Auxtel done by Y. Copin to infer its impact on the defocus. Also, it would have been interesting to upgrade this and implement the model of Von Karman and compare the results if it was not much longer.

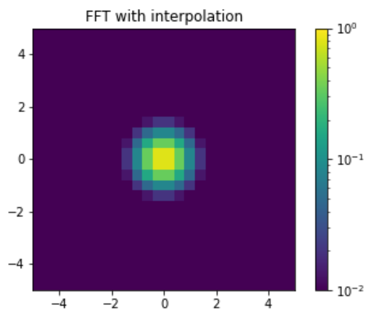
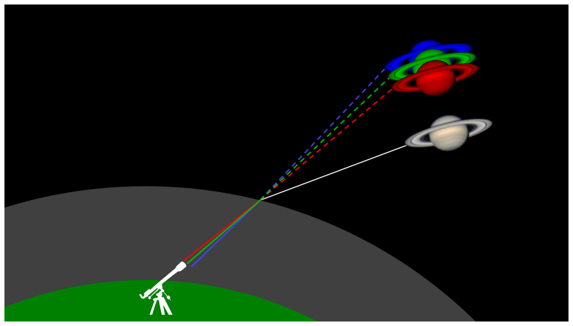


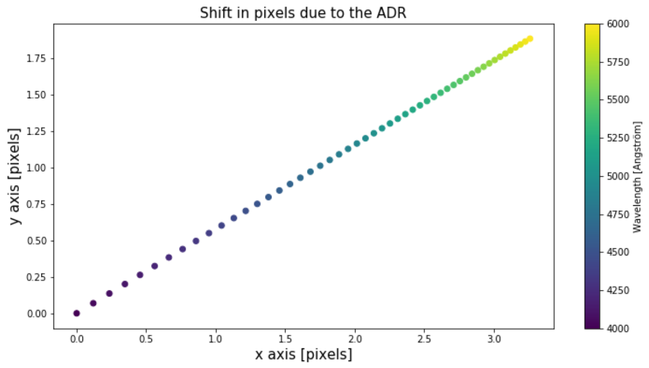
Figure 32: 2D PSF in logarithmic scale done with the interpolation of the Fourier transform of the Kolmogorov model. The results with the Hankel were not distinguishable from this one so I did not add them.

2 Atmospheric differential refraction

2.1 Phenomenon

When a light ray is emitted from a star, it goes through several layers of atmosphere which refract it. As widely shown with the prism as a diffracting object, different wavelength are refracted with different angles. As the atmosphere behaves as a diffracting object, the position at which we seem to see the star is not its real position depending on the wavelength; as shown in figure 33a. The angle at which the image is shifted is called the parallactic angle. Since refraction depends on the wavelength considered, when the light from a star is diffracted, the spectrogram obtained is shifted as shown in figure 33b. This effect is called atmospheric differential refraction. If the integration time is long, the ADR is also going to shift with time which will spread


(a) Image describing the effect of the atmospheric differential refraction. We can see Saturn's position as seen by different wavelength. Image imported from www.skyinspector.co.uk/home



(b) Example of the shift in pixels due to the ADR for an object with a parallactic angle of -60.02°and an airmass of 1.917. The color correspond to the wavelength for which there is the shown shift.

Figure 33: Atmospheric differential refraction

out the spectrogram. If the field of view is big, the ADR will also vary with space. However, as the integration time will be short and the field of view is small enough, both these effects were neglected here.

2.2 Mathematical description

As said previously, the parallactic angle η is needed in order to compute the ADR. Having the latitude ϕ, the hour angle h, the zenithal distance d_z and the declination of the target δ, it can be computed with equations (27) and (28).

$$\sin(\eta) = \frac{\sin(h)\cos(\phi)}{\sin(d_z)} \tag{27}$$

$$\cos(\eta) = \frac{\cos(\delta)\sin(\phi) - \cos(\phi)\sin(\delta)\cos(h)}{\sin(d_z)} \tag{28}$$

Also, the amplitude of the refraction is given by equation (29) when the plane-parallel approximation is made, which is coherent for normal observations conditions (when the airmass is inferior to 5).

$$R = \frac{n^2 - 1}{2n^2}\tan(d_z) \tag{29}$$

Having the parallactic angle and the amplitude of the refraction, the shift can be easily computed with equations (30) and (31), x_0 and y_0 being the original position of the image along both axis.

$$x = x_0 - R\sin(\eta) \tag{30}$$

$$y = y_0 + R\cos(\eta) \tag{31}$$

This effect can be added by including it in the dispersion law of the grating as shown in equation (32).

$$\Delta(\lambda) = \Delta_{grating}(\lambda) + \Delta_{ADR}(\lambda) \tag{32}$$

2.3 Modelling

To model the effect of the ADR, I used an algorithm developed by Y. Copin which implemented the functions introduced in section 2.2. I adapted this library to automatically return the shifts due to the ADR from a *.fits* file. As said previously, to do this the parallactic angle had to be computed. By comparing the value computed with results from the function *parallactic_angle* from astroplan [31], I saw that it gave a different result. It is because astroplan did account for the fact that the telescope could be either in the northern or in the southern atmosphere and that it could induce a change. However, this only implied a shift of the sign on the value of the ADR. As the absolute value was considered, it had no impact on this study. Furthermore, this sign problem was accounted for in the following functions to compute the ADR. Thus, in the end, the result was correct and the ADR could be negative. Afterwards, having systematised the output and the saving of the ADR, I applied this code on 148 observations made with the CTIO telescope and took out several statistics shown in table 2. On figure 34 is shown a typical example of the effect of the ADR on an observation made with the 0.9m CTIO telescope with the shift in arcsec on the ordinate and the corresponding wavelength on the ordinate. The shift in x is on average 0.513 arcsec (which correspond roughly to 1.28 pixels) and has a maximum of 1.32 arcsec (3.23 pixels). This effect is not negligible and I developed a code to implement it in the library called Spectractor (developed for Auxtel) containing simple functions to output the ADR in x and y and a Jupyter-notebook working example to show how to use it properly.

Axis	x	y
Minimum	0.01	0.038
Maximum	1.320	1.055
Average	0.513	0.247
Median	0.434	0.102
Standard deviation	0.398	0.307

Table 2: Several statistics concerning the ADR for a typical atmosphere with wavelengths from 4000Å to 6000Å. The values are in arcsec and the resolution is 0.401 arcsec/pixels.

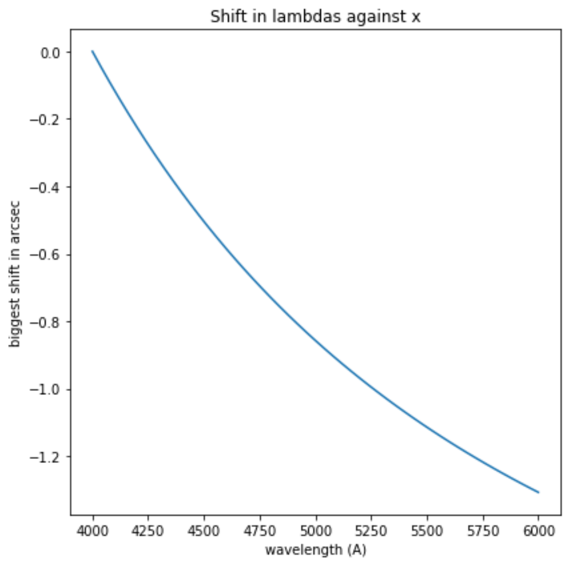


Figure 34: Example of the amplitude of the effect of the ADR with an observation made with CTIO. On the ordinate is the shift induced in arcsec and on the abscissa is the wavelength.

3 Noises

There are also several noises in observations which must be considered when trying to limit the sources of error:

- Poisson noise (or shot noise): statistical fluctuation of the flux received.
- Read-out noise: errors in the conversion and amplification of the photons received.
- Bias: voltage offset from the CCD.
- Dark current: noise due to the thermal emission of the detector.
- Cosmic rays: highly relativistic particles traversing the field of view leaving a trace.

All those noise are taken into account in the *pre-processing* and are consequently not dealt with here. There are other noises, for example the CCD induces a quantum noise. I only mentioned some noises amongst others but I did not list all the noises present in the observations.

Conclusion

In this study, I looked particularly at the effect of the defocus and of the atmospheric turbulence on the PSF and at the atmospheric differential refraction. Concerning the defocus, as advised I tried to use libraries to model it. In a first time, I tried to model a lens with HCIPy but I did not manage to make it work. Therefore, I used Zernike polynomials and it seemed to give coherent results. I also used in parallel POPPy to compare the results of both libraries. POPPy had a function *psf_thinlens* which I thought would model a lens but I found out that Zernike polynomials were used in the source code to model the lens. Afterwards, by comparing both libraries, several problems were found out. HCIPy gave a result in λ/D units whereas POPPy gave it in arcsec. By comparing the results and knowing that for the parameters used $1\lambda/D \sim 0.17 arcsec$ I concluded that there was an error with HCIPy and it was actually in arcseconds. Nonetheless, there was still a discrepancy as to match the intensity curves in focus, the x-axis of HCIPy had to be multiplied by 0.95. Furthermore, for the same input parameters and the same defocus value, both libraries did not give the same result at all. The defocus was not effective on the same ranges of longitudinal shift and, in addition to that they did not modify the PSF in the same way. It is hard to conclude on which library is the most adapted and if they work but a test of the accuracy of the result seems to be validating POPPy. However, it should be tested in more depth as I did not have the time to precisely run this test and just compared images broadly. Another noteworthy solution would be to use 2D Fourier transform and model a lens in Fourier space as there is an analytic expression describing it. Alas, I did not have the time to implement it as I spent the most time on HCIPy and POPPy.
The atmospheric turbulence could also be implemented using two libraries: HCIPy and Soapy. Nonetheless, as the computational time was too slow, they were not deemed adapted and direct use of the optical transfer function was made. The following result was in reasonable agreement with the HCIPy simulation for very low computation time. The only thing left to do is to implement it in an iterative code for the Auxtel. Also, it would be interesting to compare the computation time as there are four ways to implement it. Furthermore, it is possible to improve this model by using the Von-Karman model. Implementing this model and comparing its computational speed and its precision with the Kolmogorov model would be an interesting result.
Finally, the atmospheric differential refraction was successfully modelled and gave coherent results when compared with values computed in [1]. I developed codes to implement it with several utility functions and a Jupyter-notebook to show how these functions could be used and it is currently being implemented in an LSST library called Spectractor.
To conclude, the defocus must be further studied to compare and be able to judge if the result computed is correct or not. Both the atmospheric turbulence and the atmospheric differential refraction were successfully modelled and an interesting study would be to compare the Von Karman optical transfer function to the Kolmogorov optical transfer function to infer which one is the optimal solution in the framework of the Auxtel.

Appendix

The code

Skills developed

During this internship, I learned how to use Git (and Github and Gitlab), Jupyter, I became familiar with several python libraries, tried to optimise the computation time of the codes by vectorising them and developed several documented codes to model the effects studied here. I also developed the whole codes in Python and thus learned a lot about it and its subtleties. To make a user-friendly code, I also built utility codes documented where I detailed what the main functions were doing at the beginning of the file and added for each one more details locally. I also did input test functions to try to avoid bugs for example because of arguments. Furthermore, I raised various issues on Github and more generally on libraries, giving solutions to fix them.

Automation of the codes

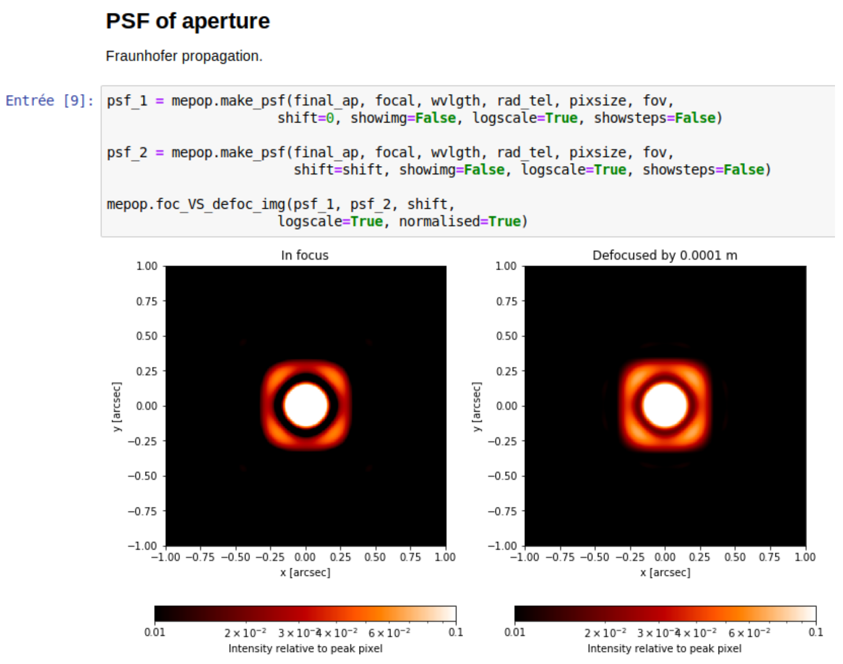
In order to do most things I did in this project more efficiently and quickly, I built a lot of functions with adapted arguments to automatise them. For example, below is shown a piece of code made to automatically build the aperture and the PSF corresponding to a given aperture with POPPy.

```python
def make_psf(final_ap, focal, wvlgth, rad_tel, pixsize, fov, shift=0,
            showimg=True, logscale=True, showsteps=False):

    osys = pop.OpticalSystem()
    osys.add_pupil(final_ap)
    if shift!=0:
        nbwaves = compute_nwaves(focal, wvlgth, rad_tel, shift)     # Zernike sign convention inversed
        osys.add_pupil(pop.ThinLens(nwaves=nbwaves,                 # Defocus in wvlgth
                                    reference_wavelength=wvlgth, radius=rad_tel))
    osys.add_detector(pixelscale=pixsize, fov_arcsec=fov)           # Add detector to optical system
    if showimg:
        if showsteps:
            psf = osys.calc_psf(wvlgth, display_intermediates=True)
        else:
            psf = osys.calc_psf(wvlgth)
            one_plot(psf, shift, logscale=logscale, normalised=True)
    else:
        psf = osys.calc_psf(wvlgth)

    return psf


def make_aperture(rad_obs, nb_spid, spid_width, rad_tel, showimg=False, rotation=45):

    bigpup = pop.CircularAperture(radius=rad_tel)
    anti_ap = pop.SecondaryObscuration(secondary_radius=rad_obs, n_supports=nb_spid,
                                       support_width=spid_width, rotation=45)
    optics=[bigpup, anti_ap]
    final_ap = pop.CompoundAnalyticOptic(opticslist=optics, name='Auxtel')
    if showimg:
        final_ap.display(npix=1024, colorbar_orientation='vertical')

    return final_ap
```

I also made functions to plot several graphs together and to be able to compare them more efficiently, be able to switch between logarithmic and linear scale or plotting the slice or the sum of the PSF. This was also really useful as it made the notebooks more readable.

Examples of utilisation

I joined two uses of the automatic codes in a Jupyter notebook. On figure 35 is shown how all the functions of POPPy were condensed to directly get the PSF and their plot. On 36 is shown an automatic plot with possible parameters such as a scale being either linear or logarithmic, the values being normalised or not and the plot being the slice or the sum of the PSF.



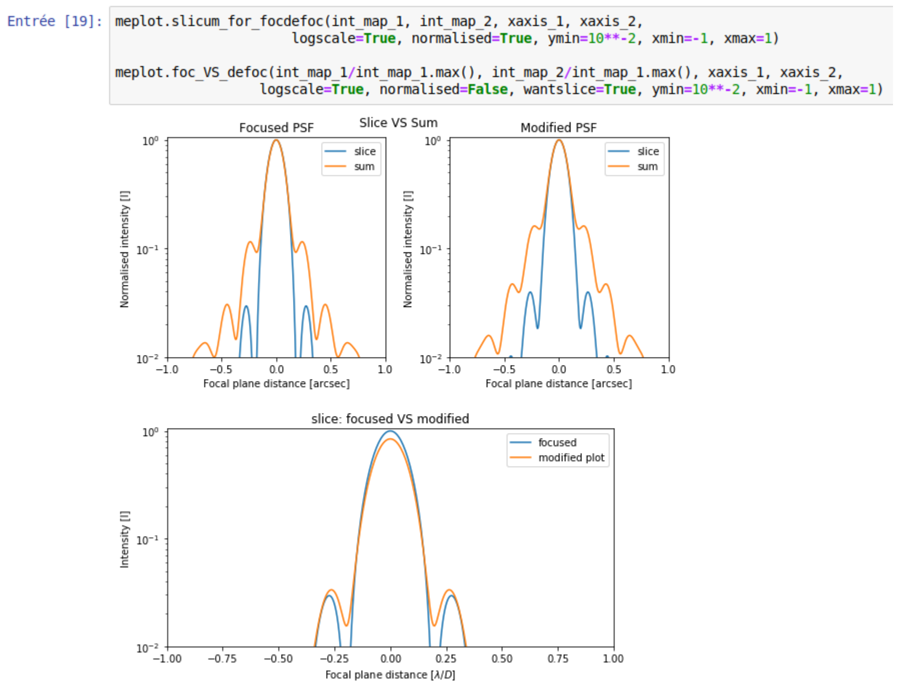
PSF of aperture

Fraunhofer propagation.

Entrée [9]:

```
psf_1 = mepop.make_psf(final_ap, focal, wvlgth, rad_tel, pixsize, fov,
                       shift=0, showimg=False, logscale=True, showsteps=False)

psf_2 = mepop.make_psf(final_ap, focal, wvlgth, rad_tel, pixsize, fov,
                         shift=shift, showimg=False, logscale=True, showsteps=False)

mepop.foc_VS_defoc_img(psf_1, psf_2, shift,
                       logscale=True, normalised=True)
```

Figure 35: Snapshot of an automated code to build and plot the PSF.



Entrée [19]:

```
meplot.slicum_for_focdefoc(int_map_1, int_map_2, xaxis_1, xaxis_2,
                           logscale=True, normalised=True, ymin=10**-2, xmin=-1, xmax=1)

meplot.foc_VS_defoc(int_map_1/int_map_1.max(), int_map_2/int_map_1.max(), xaxis_1, xaxis_2,
                    logscale=True, normalised=False, wantslice=True, ymin=10**-2, xmin=-1, xmax=1)
```

Figure 36: Snapshot of an automated code to plot the PSF with several possibilities.

References

[1] Y. Copin, *Spectro-photométrie à champ intégral dans le cadre du projet " The Nearby Supernova Factory "*. PhD thesis, Institut de Physique Nucléaire de Lyon, Université Claude Bernard Lyon 1, Aug. 2013.

[2] G. B. Brammer, P. G. van Dokkum, M. Franx, M. Fumagalli, S. Patel, H.-W. Rix, R. E. Skelton, M. Kriek, E. Nelson, K. B. Schmidt, R. Bezanson, E. da Cunha, D. K. Erb, X. Fan, N. Förster Schreiber, G. D. Illingworth, I. Labbé, J. Leja, B. Lundgren, D. Magee, D. Marchesini, P. McCarthy, I. Momcheva, A. Muzzin, R. Quadri, C. C. Steidel, T. Tal, D. Wake, K. E. Whitaker, and A. Williams, "3d-HST: a wide field grism spectroscopic survey with the Hubble Space Telescope," *The Astrophysical Journal Supplement Series*, vol. 200, p. 13, June 2012.

[3] S. Moradi, P. Moallem, and M. F. Sabahi, "Scale-space point spread function based framework to boost infrared target detection algorithms," *Infrared Physics & Technology*, vol. 77, pp. 27–34, July 2016.

[4] V. I. Tatarskii, *Wave Propagation in Turbulent Medium.* McGraw-Hill, 1961.

[5] D. L. Fried, "Optical Resolution Through a Randomly Inhomogeneous Medium for Very Long and Very Short Exposures," *Journal of the Optical Society of America*, vol. 56, p. 1372, Oct. 1966.

[6] R. Racine, "The Telescope Point Spread Function," *Publications of the Astronomical Society of the Pacific*, vol. 108, p. 699, Aug. 1996.

[7] A. Tokovinin, "From Differential Image Motion to Seeing," *Publications of the Astronomical Society of the Pacific*, vol. 114, pp. 1156–1166, Oct. 2002.

[8] M. D. Perrin, R. Soummer, E. M. Elliott, M. D. Lallo, and A. Sivaramakrishnan, "Simulating point spread functions for the James Webb Space Telescope with WebbPSF," *Space Telescopes and Instrumentation 2012: Optical, Infrared, and Millimeter Wave*, vol. 8442, p. 84423D, Sept. 2012.

[9] J. D. Long, M. D. Perrin, and R. P. Van Der Marel, "Simulating PSFs for WFIRST and JWST with WebbPSF," *American Astronomical Society Meeting Abstracts #227*, vol. 227, p. 113.01, Jan. 2016.

[10] X. Fan, "Opticspy," *http://opticspy.org/*, 2016.

[11] A. Reeves, "Soapy: an adaptive optics simulation written purely in Python for rapid concept development," *Adaptive Optics Systems V*, vol. 9909, p. 99097F, July 2016.

[12] E. H. Por, S. Y. Haffert, V. M. Radhakrishnan, D. S. Doelman, M. van Kooten, and S. P. Bos, "High Contrast Imaging for Python (HCIPy): an open-source adaptive optics and coronagraph simulator," *Adaptive Optics Systems VI*, vol. 10703, p. 1070342, July 2018.

[13] M. D. Perrin, A. Sivaramakrishnan, C.-P. Lajoie, E. Elliott, L. Pueyo, S. Ravindranath, and L. Albert, "Updated point spread function simulations for JWST with WebbPSF," *Space Telescopes and Instrumentation 2014: Optical, Infrared, and Millimeter Wave*, vol. 9143, p. 91433X, Aug. 2014.

[14] Astropy Collaboration, T. P. Robitaille, E. J. Tollerud, P. Greenfield, M. Droettboom, E. Bray, T. Aldcroft, M. Davis, A. Ginsburg, A. M. Price-Whelan, W. E. Kerzendorf, A. Conley, N. Crighton, K. Barbary, D. Muna, H. Ferguson, F. Grollier, M. M. Parikh, P. H. Nair, H. M. Unther, C. Deil, J. Woillez, S. Conseil, R. Kramer, J. E. H. Turner, L. Singer, R. Fox, B. A. Weaver, V. Zabalza, Z. I. Edwards, K. Azalee Bostroem, D. J. Burke, A. R. Casey, S. M. Crawford, N. Dencheva, J. Ely, T. Jenness, K. Labrie, P. L. Lim, F. Pierfederici, A. Pontzen, A. Ptak, B. Refsdal, M. Servillat, and O. Streicher, "Astropy: A community Python package for astronomy," *Astronomy and Astrophysics*, vol. 558, p. A33, Oct. 2013.

[15] A. Collaboration, "The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package," *The Astronomical Journal*, vol. 156, p. 123, Sept. 2018.

[16] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The NumPy Array: A Structure for Efficient Numerical Computation," *Computing in Science & Engineering*, vol. 13, pp. 22–30, Mar. 2011.

[17] E. Jones, T. Oliphant, and P. Peterson, "SciPy: Open Source Scientific Tools for Python," Jan. 2001.

[18] J. D. Hunter, "Matplotlib: A 2d Graphics Environment," *Computing in Science & Engineering*, vol. 9, pp. 90–95, May 2007.

[19] F. Pérez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," *Computing in Science & Engineering*, vol. 9, pp. 21–29, May 2007.

[20] J. Aurélien, *Développement d'un modèle numérique de l'instrument MUSE/VLT (Multi Unit Spectroscopic Explorer / Very Large Telescope).* PhD thesis, Institut National des Sciences Appliquées de Lyon, 2008.

[21] X. Gnata, *Développement d'un simulateur de performances pour le spectrographe NIRSpec du futur télescope spatial JWST.* PhD thesis, Université Claude Bernard - Lyon 1, Dec. 2007.

[22] F. A. Jenkins and H. E. White, *Fundamentals of Optics.* Fundamentals of Optics (3rd.ed.), McGraw-Hill 1957, 1957.

[23] M. Born and E. Wolf, *Principles of Optics.* Principles of Optics (4th.ed.), Pergamon Press 1970, 1999.

[24] V. Lakshminarayanan and A. Fleck, "Zernike polynomials: a guide," *Journal of Modern Optics*, vol. 58, pp. 545–561, Apr. 2011.

[25] W. Swantner and W. W. Chow, "Gram–Schmidt orthonormalization of Zernike polynomials for general aperture shapes," *Applied Optics*, vol. 33, p. 1832, Apr. 1994.

[26] F. Dai, X. Wang, and O. Sasaki, "Orthonormal polynomials for annular pupil including a cross-shaped obstruction," *Applied Optics*, vol. 54, p. 2922, Apr. 2015.

[27] D. Malacara, ed., *Optical shop testing.* Wiley series in pure and applied optics, Hoboken, N.J: Wiley-Interscience, 3rd ed ed., 2007.

[28] TelescopeOptics, "Defocus aberration," *https://telescope-optics.net/defocus1.htm*, 2006.

[29] Y. Mejia, "Exact relations between wave aberration and the sagitta difference, and between ray aberration and the slope difference," *Optik*, vol. 123, pp. 726–730, Apr. 2012.

[30] R. Kingslake and R. R. Shannon, eds., *Applied optics and optical engineering. Vol. 11: ...* New York: Acad. Press, 1992. OCLC: 831340626.

[31] B. M. Morris, E. Tollerud, B. Sipőcz, C. Deil, S. T. Douglas, J. Berlanga Medina, K. Vyhmeister, T. R. Smith, S. Littlefair, A. M. Price-Whelan, W. T. Gee, and E. Jeschke, "astroplan: An Open Source Observation Planning Package in Python," *The Astronomical Journal*, vol. 155, p. 128, Mar. 2018.