CONTENTS

This document contains all the reports of my projects as follows:

1. My MSc Thesis report titled “Theoretical modeling of large-scale magnetic field reversals in spiral galaxies”. As I’m still working on my thesis, the complete report will be available by April 2026, tentatively. (**20 pages**)
2. During the Spring Semester of 2025, I worked on independently reproducing the results of the paper Basu et al. (2018) by corresponding with the first author. Here is the report of my work during our discussions on “Rotation Measure Statistics”. (**9 pages**)
3. Abstract and report of the project at RRI titled “Data Reduction Pipeline for the Removal of Polarized Radio Foreground”. [2024] (**62 pages**)
4. Report of the reading project titled “Understanding the Large Scale Magnetic Fields of Spiral Galaxies”. [2023] (**5 pages**)
5. Report of the project titled “A Study of Transiting Exoplanets using TESS Data. [2022] (**17 pages**)
6. A comprehensive course list with the relevant topics covered in the above, with the credits, grades and textbooks used.

In all the above-mentioned, Shishir H and Shishir Halaharivi refer to me.

I hereby confirm that the information provided in this document is true and accurate to the best of my knowledge.

Thank you very much.

Reversals of the large-scale magnetic fields of galaxies and their evolution over cosmic time

Submitted By

H Shishir

School of Physical Sciences

National Institute of Science, Education and Research (NISER), Bhubaneswar

Under the Guidance of

Dr. Luke Chamandy

Reader - F

School of Physical Sciences

National Institute of Science, Education and Research (NISER), Bhubaneswar

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Dr. Luke Chamandy, for his continuous guidance, support, and encouragement throughout the course of this project. His insightful suggestions, critical feedback, and patient explanations have been invaluable in shaping both the scientific content and presentation of this work. Beyond the technical aspects of the project, I am grateful for his advice on effective communication, helping me structure the report clearly and present the results confidently.

I would also like to extend my heartfelt thanks to Mr. Sukanta Ghosh for his assistance in understanding and navigating the computational code used in this study. His guidance and willingness to clarify doubts greatly facilitated my progress and deepened my understanding of the tools and methods employed.

Finally, I wish to acknowledge the support of my peers, and the academic environment at NISER, Bhubaneswar, which provided the resources and motivation necessary to complete this work. The encouragement and constructive discussions I received throughout this journey have been instrumental in making this project both productive and enjoyable. I am also deeply thankful to my parents for their unwavering support and encouragement throughout my academic journey.

H Shishir
5th Year Integrated M.Sc.
School of Physical Sciences
NISER, Bhubaneswar

Abstract

Large-scale magnetic field reversals in spiral galaxies remain among the least understood aspects of cosmic magnetism. While a reversal is firmly established locally in the first quadrant of the Milky Way and only tentatively detected in the external galaxy NGC 4666, recent high-resolution cosmological simulations suggest that such reversals may be more common than previously thought. In this work, we investigate whether the apparent rarity of reversals is intrinsic or simply a consequence of observational limitations. Our study predicts that the fraction of galaxies exhibiting reversals at any redshift can be as low as 5%, which is low but still higher than what current observations seem to suggest. This contrast highlights both the observational challenges involved and the need for large statistical samples to accurately determine the true occurrence and frequency of magnetic field reversals.

Using a combination of semi-analytic galaxy formation models (GALFORM) and galactic dynamo simulations (MAGNETIZER), we examine over a million galaxies and track the evolution of both turbulent and large-scale magnetic fields. Two dynamo prescriptions, Rodrigues et al. (2019) and Jose et al. (2024), are employed to predict the occurrence, strength, asymmetry, and radial distribution of magnetic field reversals. Our analysis shows that reversals are widespread in simulated galaxies, with the majority exhibiting weak and asymmetric reversals, though stronger, near-symmetric events are occasionally observed. The radial locations of reversals are broadly distributed across galactic disks, suggesting that reversals can occur at nearly any radius.

We find that reversals arise across a wide range of galactic environments, with their prevalence and properties strongly dependent on the dynamo model. R19 predicts reversals that become more common toward late cosmic times, often with multiple reversals, whereas J24 favours early, predominantly single reversals. By tracking the radial motion of reversals across snapshots and merger events, we measure reversal propagation speeds of order 0.1–0.4 km s^{-1}, consistent with theoretical expectations for dynamos with finite turbulent diffusivity. A moderate correlation is found between reversal speed and radial position, with outward-moving reversals tending to accelerate toward the disk edge.

Finally, we investigate the observational signatures of reversals using synthetic Rotation Measure (RM) maps, demonstrating that a single reversal in an axisymmetric disk produces a characteristic sinusoidal RM pattern. This highlights the potential for detecting and characterizing magnetic field reversals in real galaxies through Faraday rotation measurements.

Our results offer the first statistical view of reversal formation in semi-analytic dynamo models, highlight their sensitivity to galactic assembly histories, and provide concrete diagnostics for detecting such features in future radio observations.

Contents

Acknowledgement **i**

Abstract **ii**

Page

1 Why Study Reversals? **2**

1.1 Introduction 2

1.2 Galaxy Formation and Dynamo Simulations 3

2 Detecting Reversals in our Simulation **4**

2.1 Choosing the right Galaxies 4

2.2 Quantitative Measures to characterize Reversals 5

2.2.1 Strength: 5

2.2.2 Asymmetry 6

2.2.3 Location 7

3 Galaxies sorted according to number of reversals **7**

3.1 Galaxies are sorted by number of Reversals 7

3.2 Determining the Speed of Reversals 7

3.3 Correlations of Speed of Reversals with Galaxy and Dynamo Properties 9

4 Rotation Measure **9**

5 Conclusion and Future Work **11**

5.1 Key Results and Conclusion 11

5.2 Future Work 11

6 Appendix **12**

6.1 Derivation of the Thin–Disc Mean–Field Dynamo Equations 12

6.2 Derivation of the Dimensionless control parameters 13

6.3 Total number of galaxies that exhibit reversals 14

6.4 An intriguing distribution - D/D_c 14

6.5 Correlation between velocity of reversal and Halo mass 14

1 Why Study Reversals?

1.1 Introduction

Magnetic fields are ubiquitous across cosmic structures—from stars and galaxies to the intergalactic medium—yet their origin, amplification mechanisms, and evolution are still not fully understood. In this work, we focus on magnetic field structure in spiral galaxies. Galactic magnetic fields play a crucial role in shaping various astrophysical processes: they influence star formation by regulating the collapse of molecular clouds, affect cosmic ray propagation and confinement, and contribute to the stability and dynamics of the interstellar medium (ISM). Moreover, they can impact gas accretion and outflows, thereby influencing the overall evolution of galaxies.

A magnetic field reversal occurs when the direction of the magnetic field vector changes by approximately 180^0 as we move radially across the galactic disk. One of the most well-studied examples is found in our own Milky Way Galaxy [9, 17, 15, 7, 21, 19]. There is ample observational evidence suggesting that the nearest reversal lies about 600 pc from the Sun, located roughly 7.5 kpc from the Galactic Center. This reversal is detected along the Carina–Sagittarius arm, where the magnetic field is oriented anticlockwise with respect to the North Galactic Pole as shown in 1.1. In contrast, the magnetic field in the regions away from this arm is directed clockwise and points inward toward the Galactic Center. There have been conflicting studies regarding whether this magnetic field reversal extends throughout the spiral arm. However, observations do provide solid evidence for a localized reversal in the first Galactic quadrant, suggesting that at least part of the disk hosts a well-defined reversal in the magnetic field direction [12, 4, 3, 8].

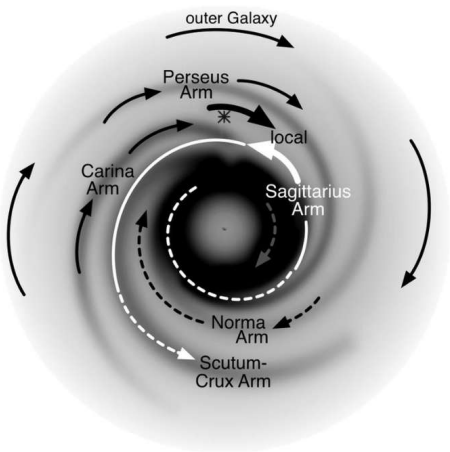
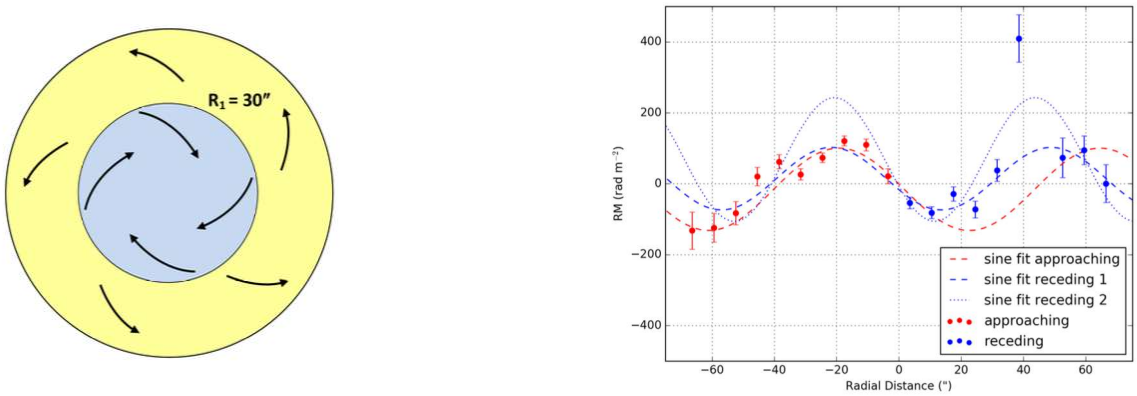


Fig. 1.1: Artist depiction of large scale magnetic field structure of Milky Way depicted by the arrows. "*" represents the position of the sun

Naturally, this raises the question of whether the Milky Way is unique in exhibiting such reversals. For a handful of galaxies for which the large scale magnetic fields have been well resolved only one galaxy shows some evidence of such a reversal. The spiral starburst galaxy NGC 4666, for instance, shows indications of a large-scale magnetic field reversal. Using Rotation Measure (RM) Synthesis, Y. Stein (2019) reported that the magnetic field in the inner 5 kpc of the disk appears to be oriented inwards and clockwise, while beyond this radius the field seems to flip direction, becoming outwards and anticlockwise as shown in 1.2 [20]. However, the authors also note that this interpretation is somewhat tentative, as the observational constraints are limited and the inferred reversal depends sensitively on the modelling assumptions.

These reversals are of particular interest as they offer valuable insights into the underlying galactic dynamo mechanisms, aid in developing accurate models for the evolution of large-scale magnetic structures, and may also provide clues about a galaxy's interaction and merger history. Galaxies that have experienced minor or major mergers, or are gravitationally interacting with satellite companions, often exhibit complex or disturbed magnetic configurations. Interestingly, both NGC 4666 [23, 11] and the Milky Way are gravitationally interacting systems, each with nearby companions and evidence of past mergers. This similarity hints at a possible connection between galactic interactions and the occurrence of magnetic field reversals, a link that our study seeks to explore further.



(a) **(b)**

Fig. 1.2: (a) The predicted large scale magnetic field structure of NGC 4666. (b) The Rotation Measure as a function of radial distance where the sinusoidal fit was made for the different sides separately indicated by the color (sine fit for receding 2 include all the points including the outlier).

Recent advances in cosmological galaxy simulations have made it possible to follow the growth and evolution of magnetic fields within galaxies and their surrounding halos [14, 6, 3]. Newer high-resolution zoom simulations look at a broader range of disc galaxies as indicated in Pakmor (2023) and depicted in 1.3 [16]. Analysis of these simulated galaxies reveals an intriguing pattern: maps of the azimuthal magnetic field at the present epoch show that nearly all of them develop at least one large-scale magnetic field reversal. Only one of the simulated systems departs from this trend, indicating that such reversals may be a common outcome of galaxy formation in cosmological environments.

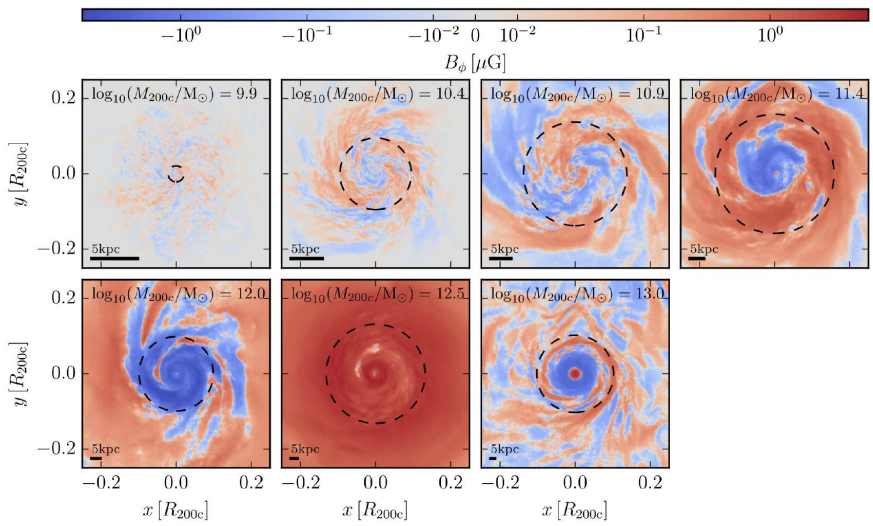


Fig. 1.3: Azimuthal component of the magnetic field of the cosmological galaxies at z = 0.

1.2 Galaxy Formation and Dynamo Simulations

We employ the GALFORM semi-analytic model of galaxy formation (SAMGF) to follow the evolution of more than a million galaxies spanning the stellar mass range $10^{12} - 10^{14}$ $M_\odot$. GALFORM extracts the assembly histories of dark matter haloes from a cosmological simulation in the form of merger trees, and then evolves the baryonic components by solving differential equations that describe star formation and its regulation. This yields a comprehensive catalogue of global galaxy properties—such as disc and bulge masses, gas content, and half-mass radii—at any redshift. Building on these outputs, we use MAGNETIZER to couple galactic dynamo theory with the SAM framework, allowing us to predict the evolution of both turbulent and large-scale magnetic

fields as a function of time and galactocentric radius. For clarity and to avoid unnecessary complexity, all galaxy properties in this combined modeling approach are assumed to be axisymmetric.

To study magnetic field reversals in galaxies, we make use of two MAGNETIZER models: the framework of Rodrigues et al. (2019), hereafter **R19** [18], and the more recent model of Jose et al. (2024), hereafter **J24** [10]. Few key differences between each of the models is described in Table 1.1. f_B is defined as the fraction of the interstellar turbulent energy density that is carried by the random magnetic field, v_t is the turbulent velocity, R_κ is a dimensionless control parameter, f_{seed} scale factor for seed field and SFR is the Star Formation Rate.

Table 1.1: Comparison between R19 and J24

	R19	**J24**
f_B	0.5	1.0
v_t	10 km/s	25 km/s for $SFR < 3\,M_\odot/yr$ otherwise $v_t(SFR)$
R_κ	0.3	1.5
f_{seed}	10^{-3}	10^{-2}

Table 1.2: Percentage of Galaxies with reversal as a function of redshift (z)

z	**R19**(%)	**J24** (%)
2	3.35	0.1
1	3.18	0.07
0	3.01	0.09

2 Detecting Reversals in our Simulation

2.1 Choosing the right Galaxies

In this work, we focus on well-evolved galaxies and consider only those reversals that are physically detectable. Our selection procedure consists of two stages, performed on 40000 galaxies. A schematic is shown in fig. 2.1

.

Stage 1: Filtering unevolved galaxies;

We first remove galaxies whose magnetic fields have not grown significantly beyond the seed field. For this purpose, we require the volume-averaged total magnetic energy density, B_0, to exceed $0.05\mu G$. Galaxies that do not satisfy this condition are excluded from further analysis.

$$B_0 = \left[\frac{\int_0^{2\pi} \int_0^{r_d} \int_{-h(r)}^{h(r)} \left(B_r^2(r) + B_\phi^2(r) + B_z^2(r)\right) e^{-\frac{|z|}{h(r)}}\, r\, dr\, d\phi\, dz}{\int_0^{2\pi} \int_0^{r_d} \int_{-h(r)}^{h(r)} r\, dr\, d\phi\, dz} \right]^{1/2} \geq 0.05\mu G \tag{2.1}$$

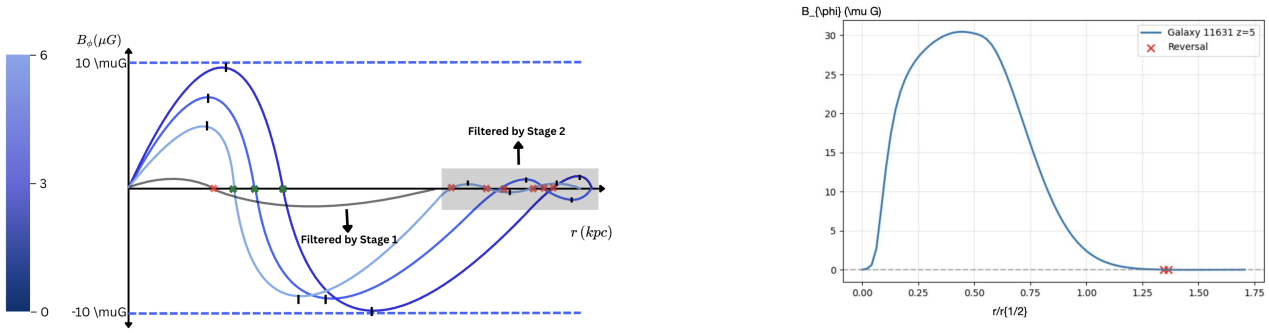
Stage 2: Detecting magnetic field reversals;

To identify reversals, we examine the azimuthal component of the magnetic field, B_ϕ at the mid-plane of the galaxy. A reversal occurs wherever B_ϕ changes sign. To locate these reversals, we begin by identifying all local extrema of $B_\phi(r)$. We then select only those pairs of consecutive extrema that have opposite signs. The reversal is obtained by cubic spline interpolating between such extrema.

Despite restricting our sample to evolved galaxies, we observed that remnants of the seed field can persist in the outermost regions. To avoid false detections caused by these low-level artifacts, we impose an additional criterion that each extremum must have a magnitude greater than or equal to the local floor field at that radius ($r_{extrema} \equiv r_{ex}$). Only extrema satisfying this condition are considered valid for the detection of reversals.

$$B_\phi(r_{ex}) \geq f(r_{ex}) = \frac{\pi^2\, \eta(r_{ex})}{4\, h(r_{ex})^2} \left(1 + \frac{R_U(r_{ex})}{\pi^2}\right) \left(1 - \frac{D(r_{ex})}{D_c(r_{ex})}\right) \widetilde{B}(r_{ex})[\mu G] \tag{2.2}$$

where $R_U(r) \equiv \frac{U_Z h(r)}{\eta_t(r)}$ is a dimensionless control parameter, $D(r) \equiv \frac{\alpha_0(r) G(r) h(r)^3}{\eta_t(r)^2}$ is the dynamo number, $D_c = -\frac{\pi^5}{32}$ is the critical Dynamo number, $\widetilde{B}(r)$ is the seed field and $\alpha_0 = \frac{l(r)^2 \Omega(r)}{h(r)}$ where Uz is the outflow velocity (zero in our case), $h(r)$ the disc half-thickness, Ω angular velocity of the gas, η_t turbulent magnetic diffusivity and $G(r)$ is the shear.



(a) (b)

Fig. 2.1: (a) A schematic of azimuthal component of the magnetic field (B_ϕ) as a function of radius for multiple redshifts. The selection procedure eliminates the reversals (reversals) in red and only the green ones are considered. (b) An example from the simulation which will be excluded.

As shown in 1.2, after the robust filtering process we notice that R19 shows higher number of galaxies with reversals but overall the fraction is less than 5 %.

2.2 Quantitative Measures to characterize Reversals

Given that magnetic field reversals remain largely unexplored and that we are among the first to examine them in detail, we aim to characterize these reversals by introducing suitable quantitative measures.

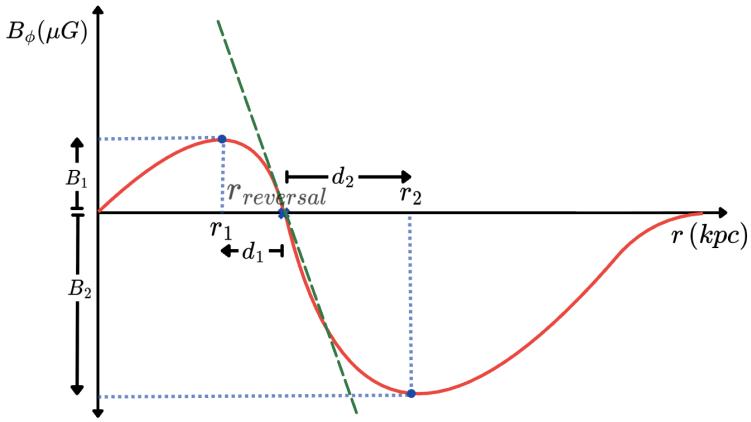


Fig. 2.2: A schematic of azimuthal component of the magnetic field (B_ϕ) as a function of radius. (B_1) is the first extrema and (B_2) is the second extrema and r_1 and r_2 are the corresponding radii of their occurance. $r_{reversal}$ refers to the point of reversal.

2.2.1 Strength:

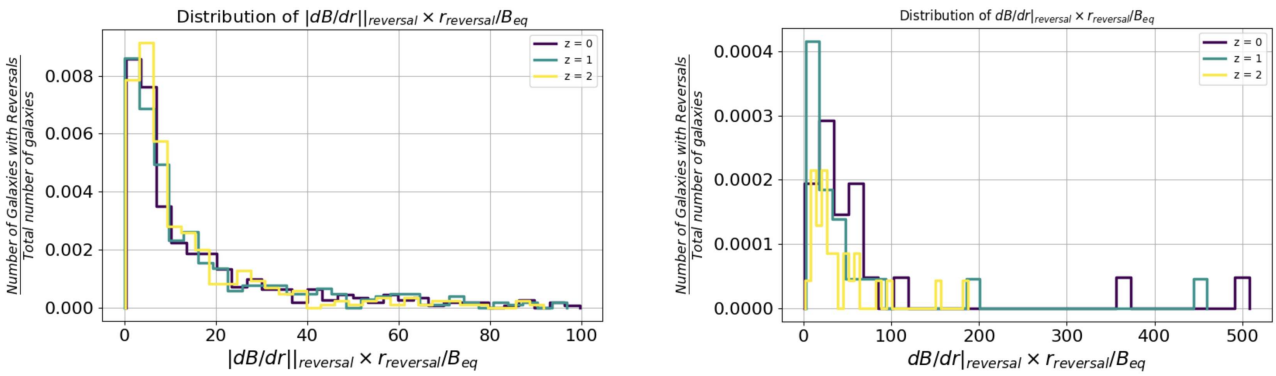
Figure 2.2 shows the azimuthal magnetic field component, B_ϕ, at a given redshift z, exhibiting a reversal at a radius $r_{reversal}$ ($\equiv r_r$), where $B_\phi = 0$. To quantify the **Strength** ($\mathcal{S}$) of the reversal, we define it as the ratio between the change in the field amplitude (from the first extrema (B_1) to the second (B_2)) and the radial separation between these extrema.

$$\mathcal{S} = \frac{B_2 - B_1}{r_2 - r_1} \tag{2.3}$$

In order to make the quantity dimensionless, we scale the slope at reversal ($\frac{dB}{dr}|_{r=r_r}$) by $r_\mathrm{r}/B_\mathrm{eq}(r_\mathrm{r})$, where we consider the ratio of the radius at reversal to the equipartition field strength at the same location. Thus **Dimensionless Strength** ($\mathcal{S}'$) is given by,

$$\mathcal{S}' = \left.\frac{dB}{dr}\right|_{r=r_r} \times \frac{r_r}{B_{eq}(r_r)} \tag{2.4}$$

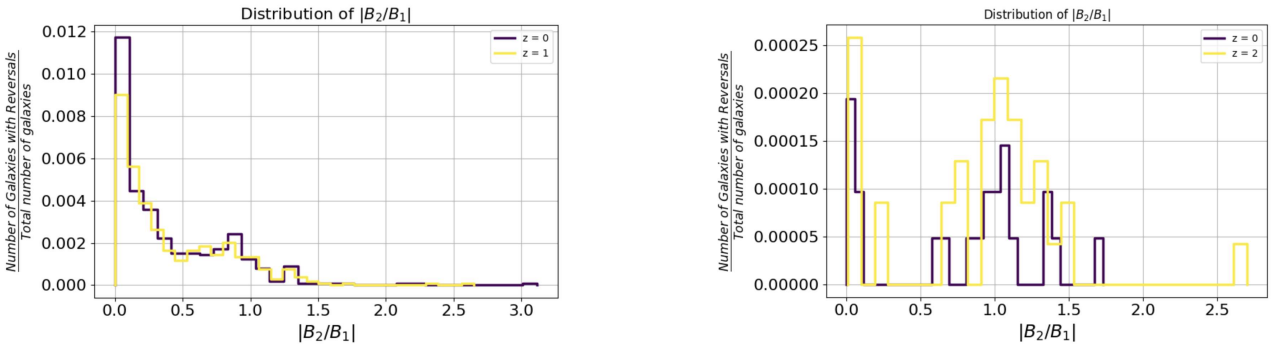
The quantity $\mathcal{S}$ measures the rate of change of the azimuthal magnetic field across the reversal, essentially quantifying how sharply the field flips direction. A larger $\mathcal{S}$ corresponds to a more abrupt reversal, while a smaller value indicates a smoother transition. This gives a first-order sense of the "intensity" or significance of the reversal in physical terms. The dimensionless strength, $\mathcal{S}'$, scales the slope at the reversal by the local radius and equipartition field, removing the dependence on the absolute field strength and galactic size.



(a) R19 **(b)** J24

Fig. 2.3: Distribution of Dimensionless Strength $\mathcal{S}'$ normalized to the number of galaxies exhibiting reversals to the total number of galaxies, at redshifts 0, 1 and 2.

In figure 2.3, the distribution of $\mathcal{S}'$ is skewed toward zero, showing that most reversals in the galaxy population are relatively weak. Only a smaller fraction of galaxies exhibit strong reversals with high $\mathcal{S}'$. This highlights that abrupt reversals are rare, whereas smooth, gradual reversals are more common.



(a) R19 **(b)** J24

Fig. 2.4: Distribution of Asymmetry $\mathcal{A}$ normalized to the number of galaxies exhibiting reversals to the total number of galaxies, at redshifts 0, 1 and 2.

2.2.2 Asymmetry

Another useful quantity is the ratio of the absolute value of second extrema to the first extrema on either side of the reversal, given by **Asymmetry** ($\mathcal{A}$),

$$\mathcal{A} = \left|\frac{B_2}{B_1}\right| \tag{2.5}$$

tells you how symmetric or asymmetric the magnetic field strength is on the two sides of the reversal.

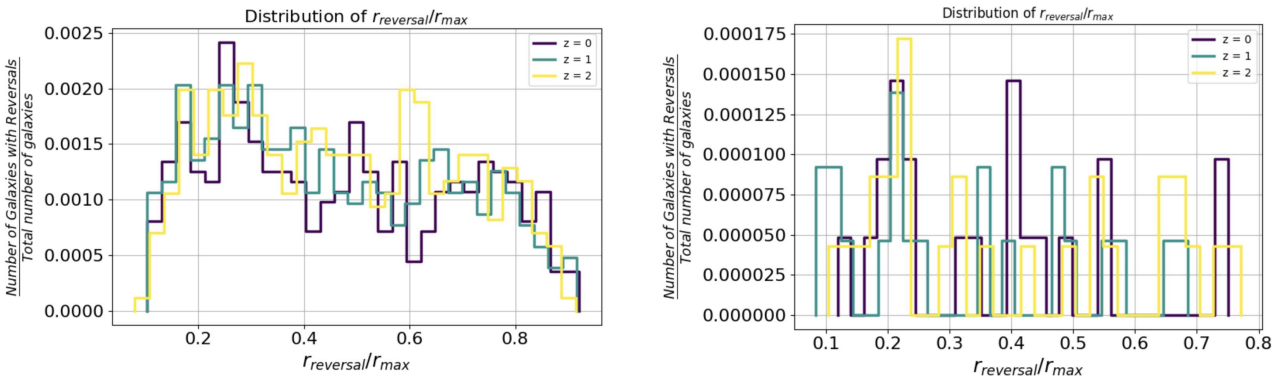
- In R19, the PDF of Asymmetry given by the figure 2.4b shows that the magnetic field strength is asymmetric on either side of the reversal, with $|B_2/B_1| < 1$ in most cases. This indicates that the first extremum is generally larger than the second, consistent with the expected radial decline in magnetic field strength.
- In J24, although the lopsidedness PDF is still skewed toward values less than 1, there is a distinct peak near 1, indicating that some reversals exhibit near-symmetry.

2.2.3 Location

We can also define the ratio of location of the reversal along the radius to the maximum radius of the galaxy, defined as $((\mathcal{R}))$,

$$\mathcal{R} = \frac{r_r}{r_{max}} \tag{2.6}$$

This quantity provides an estimate of where within the galaxy a reversal is likely to occur is given in figure 2.5. In both models, R19 and J24, the PDF of $\mathcal{R}$ is nearly flat, indicating that reversals can occur at essentially any radial position within the disk.



(a) R19 **(b)** J24

Fig. 2.5: Distribution of ratio of location of the reversal along the radius to the maximum radius of the galaxy, $\mathcal{R}$ normalized to the number of galaxies exhibiting reversals to the total number of galaxies, at redshifts 0, 1 and 2.

3 Galaxies sorted according to number of reversals

3.1 Galaxies are sorted by number of Reversals

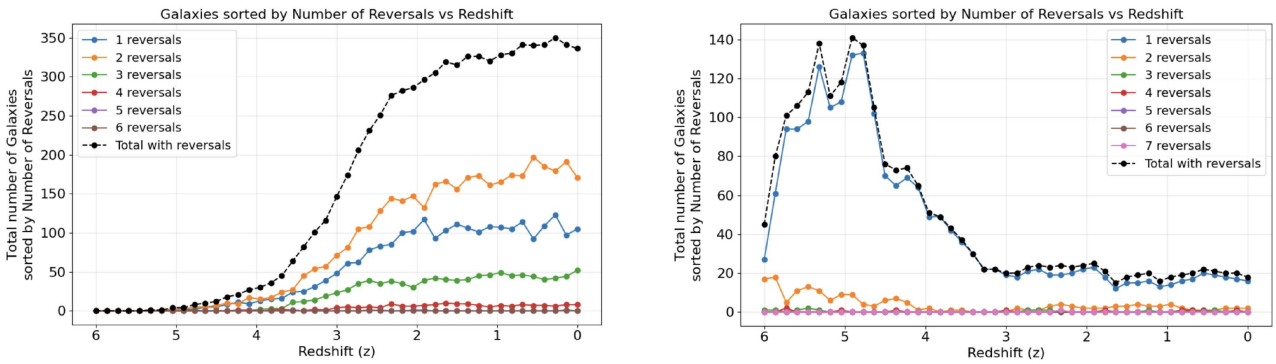
We now have a framework that allows us to track magnetic field reversals in all valid galaxies across cosmic time, enabling us to study their evolution. Figure 3.1 shows the total number of galaxies exhibiting at least one reversal at each redshift (dashed line). The solid lines indicate, for each snapshot, the number of galaxies containing a specific number of reversals. Here, we notice that

- In R19, the number of galaxies exhibiting reversals increases toward later times. Galaxies with two reversals contribute the largest fraction, followed by next to leading order by those with one and three reversals.
- In J24, by contrast, reversals are more common at earlier times, and galaxies with a single reversal dominate overwhelmingly.

3.2 Determining the Speed of Reversals

This distinction is significant, as it indicates that the occurrence of reversals are highly sensitive to the model under consideration. In the case of R19, the dynamo is weak hence the magnetic fields are slow to evolve and do not satisfy our selection criteria at early times.

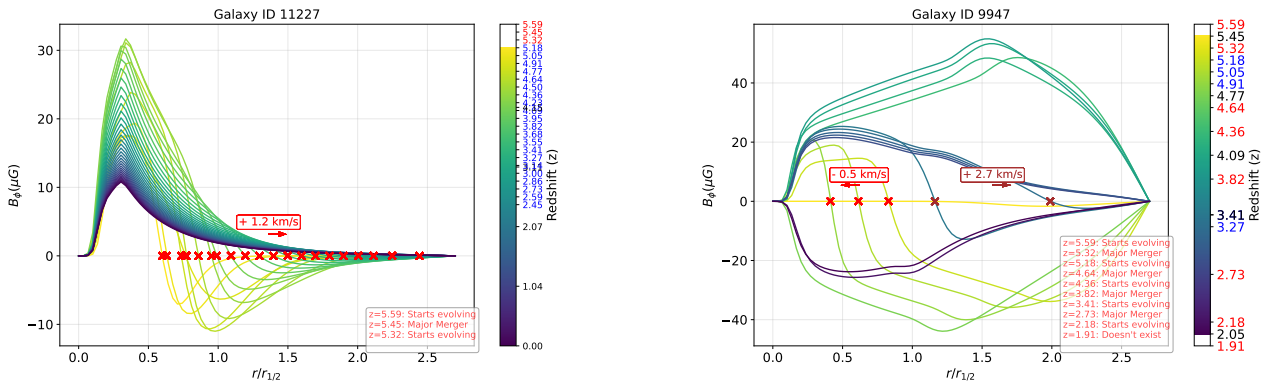
To study the evolution of reversals, we also track how the reversal moves radially over time, allowing us to estimate the reversal speed. This requires several precautions. First, we restrict the analysis to galaxies that exhibit exactly one reversal, ensuring that the radial position is well defined at each snapshot. Second, we incorporate each galaxy's merger history: if a system undergoes a merger, the reversals before and after the event are treated as separate sequences, and their dynamics are tracked independently.



(a) R19 **(b)** J24

Fig. 3.1: The evolved galaxies are sorted by number of reversals as a function of redshift(z) for (a) R19 and (b) J24

We examine two galaxies from the J24 model. In the figure 6.2b of azimuthal component of magnetic field as a function of radius, red crosses mark the locations of magnetic field reversals. On the color bar, red points indicate the evolutionary history of the galaxy, including major mergers, when the galaxy meets our criteria for being considered evolved, and when it no longer exists as an independent system and has become part of the central galaxy. Blue numbers correspond to the redshifts at which reversals are detected. Additionally, the galaxy's evolutionary history is annotated on the plot, along with the direction and speed of the reversal, providing a comprehensive view of reversal dynamics in the context of galaxy evolution.



(a) **(b)**

Fig. 3.2: Azimuthal component of the magnetic field (B_ϕ) as a function of radius normalized to half mass disk radius ($r_{1/2}$). Red crosses indicate the locations of magnetic field reversals. The color bar shows the evolutionary history of the galaxy: red points mark major mergers, the time when the galaxy meets our criteria for being considered evolved, and when it ceases to exist as an independent system and merges into the central galaxy. Blue numbers indicate the redshifts at which reversals are detected. Annotations on the plot show the galaxy's evolutionary history, including the direction and speed of the reversals.

- In Galaxy 11227, after a merger at $z = 5.45$, a reversal is generated at $z = 5.18$ and persists until $z = 2.45$, with an average speed of 1.2 km/s. The reversal propagates outward toward the edge of the galaxy before disappearing.
- In this case, as the galaxy is not interrupted by a major subsequent merger, the reversal propagates throughout the galaxy over a period of 1.6 Gyr.

- In Galaxy 9947, two separate reversal events occur before and after a merger at $z = 4.64$. The reversal prior to the merger moves inward with a velocity of -0.5 km/s, while the reversal after the merger moves outward at 2.7 km/s. Both reversals last for a few hundred Myr.
- In both cases, the reversals do not fully propagate from the centre to the edge of the galaxy or vice versa, as they are interrupted by merger events.
- Overall, in our simulations, most reversals originate after mergers and are often subsequently interrupted by additional merger events, highlighting the strong influence of galactic interactions on reversal dynamics.

Using the time evolution of the reversals using the above framework, we compute the radial speed of the reversal for all the evolved galaxies and the time period for which they last.

- For the R19 model, we obtain a mean reversal velocity of 0.4 km/s with a standard deviation of 0.3 km/s. For J24, the mean velocity is lower, 0.1 km/s, but with a substantially larger standard deviation of 1.7 km/s, reflecting a broader diversity in reversal behaviour.
- For the R19 model, the reversals last for 2.8 Gyrs with a standard deviation a 2.1Gyrs. Whereas for J24, they last for 0.9 Gyrs with a standard deviation 1.6 Gyrs.

These values are consistent with theoretical expectations. In particular, the results are of the same order of magnitude as those reported by Mikhailov and Khasaeva (2018) [13] for systems with finite turbulent diffusivity η_t.

3.3 Correlations of Speed of Reversals with Galaxy and Dynamo Properties

We observe a correlation between the speed of magnetic field reversals and their radial position from the galactic centre, considering galaxies with a single reversal in the case of R19.

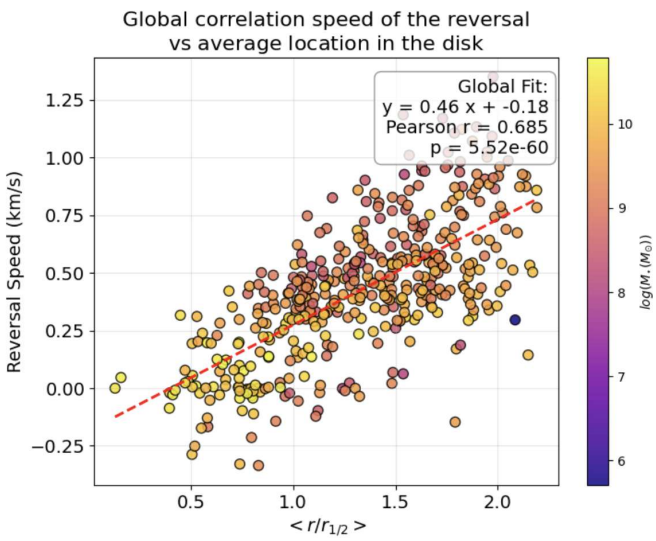


Fig. 3.3: (**Left**) Correlation of speed of the reversal (in km) against average $r/r_{1/2}$. The points are coloured with respect to the average stellar mass M_* of the galaxy as indicated by the colourbar. The red line indicates a liner fit for the scatter plot.

- The trend indicates that the reversal speed generally increases as one moves outward toward the edge of the galaxy. Reversals are more likely to move outward.
- Near the inner disk, reversals tend to move more slowly, and in some cases, propagate inward toward the galactic centre.
- The Pearson correlation coefficient is $p = 0.7$, indicating a moderate correlation between reversal speed and its radial location.
- For galaxies with high stellar mass, several cases exhibit reversal speeds that are nearly zero, suggesting that dynamics in massive systems may be influenced by additional factors such as merger history or complex magnetic field structure.

While the revelation of the result is very intriguing we are still working on understaing the physical reason behind it.

4 Rotation Measure

Faraday rotation is observed as rotation of linear polarization, caused by the different propagation speeds of left- and right-circularly polarized components experience slightly different refractive indices due to interactions with free electrons in the plasma. The amount of rotation (ψ) grows with the square of the wavelength, producing a characteristic λ^2 dependence that makes Faraday rotation especially prominent at low radio frequencies. The quantity used to describe this effect is the rotation measure (RM).

$$\psi(\lambda) = \psi_0(B_\perp) \ + \ RM \ \lambda^2 \tag{4.1}$$

where ψ_0 is the intrinsic rotation angle which is dependent on the perpendicular component of the magnetic field, $B_\perp$.

RM encapsulates the cumulative influence of the magnetized plasma along the line of sight, depending on both the electron density (n_e) and the component of the magnetic field parallel to the observer ($B_{||}$). Formally expressed as an integral over the path of the radiation, it effectively measures how much magnetic fields, weighted by the density of intervening electrons, twist the polarization angle.

$$\mathrm{RM} = 0.812 \int_0^L n_e(l)\, B_{\|}(l)\, dl \tag{4.2}$$

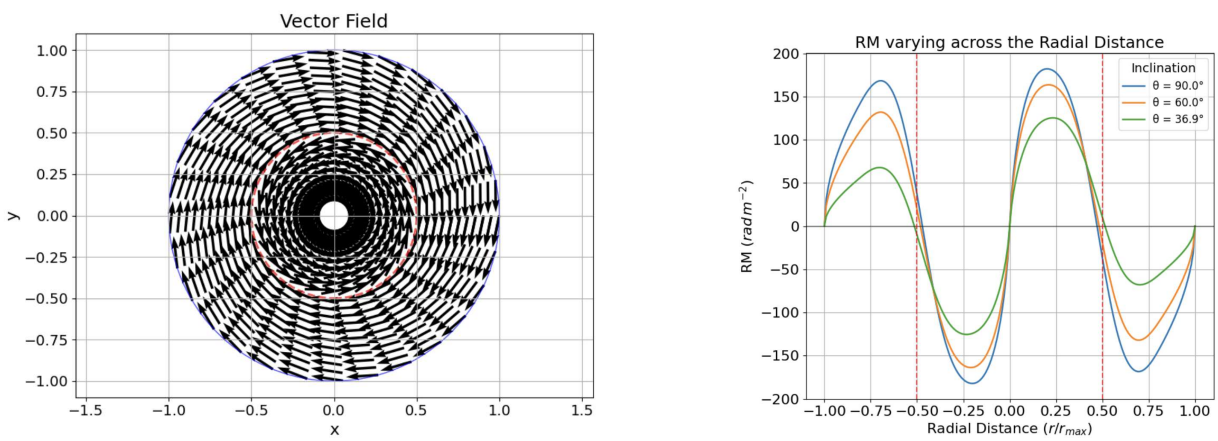
where n_e is in cm^{-3}, $B_{||}$ in μG and l is in pc.

Rotation Measure (RM) provides a powerful tool for probing the large-scale magnetic field structure of galaxies. To explore this, I have developed a code that generates synthetic magnetic field vectors across a model galactic disk. I focus specifically on galaxies with axisymmetric magnetic fields, which means that the dominant mode in the Fourier decomposition of the magnetic field is the m=0 mode. In addition, the galaxies of interest exhibit a magnetic field reversal at half the disk radius. To capture this configuration, we adopt an axisymmetric parametrization of the magnetic field in both the inner and outer disk with B_0, $p_{0(innerdisk)}$ and $p_{0(outerdisk)}$ and allowing for a smooth transition across the radius we scale the field by e^{-r/r_s}.

$$B_x = \left(B_0 \sin p\, e^{-r/r_s}\right) \cos\phi - \left(B_0 \cos p\, e^{-r/r_s}\right) \sin\phi, \tag{4.3}$$

$$B_y = \left(B_0 \sin p\, e^{-r/r_s}\right) \sin\phi + \left(B_0 \cos p\, e^{-r/r_s}\right) \cos\phi, \tag{4.4}$$

$$B_z = 0. \tag{4.5}$$



(a) **(b)**

Fig. 4.1: (a) A synthetic large scale magnetic field vector map where in the inner disc (r/r_{max}0.5) are pointing in the anti-clockwise direction and in the outer disk (r/r_{max}0.5) they point in the clockwise direction. (b) The Rotation Measure as a function of Radial Distance calculated for field vectors in (a). The red verticle line indicates the points of reversal ($r/r_{max} = 0.5$)

where n_e is in cm^{-3}, $B_{||}$ in μG and l is in pc.

with B_0, $p_{0(innerdisk)}$ and $p_{0(outerdisk)}$ being -0.5 μG, 2.79 rad and 5.59 rad.

With this model magnetic field, I then apply a line-of-sight integration tool to compute the resulting RM distribution only across the midplane (no halo contribution). By varying the inclination angle of the galaxy relative to the observer, the code produces synthetic RM maps that illustrate how a single magnetic field reversal manifests observationally.

This approach allows us to investigate how geometric effects, disk orientation, and field reversals shape the observable RM signal, providing insight into how such reversals may be detected in real galaxies.

- The RM as a function of radial distance exhibits a sinusoidal pattern. This serves as a proof of concept for why, in axisymmetric models of galactic magnetic fields, the RM is often fit with a sine curve.
- A positive RM indicates that the magnetic field points toward the observer, while a negative RM signifies the opposite. In this setup, the observer is looking along the negative y-direction.
- The magnitude of RM reflects the strength of interaction between the medium and the polarized radiation: large RMs correspond to dense gas, strong magnetic fields, or long path lengths along the line of sight.

5 Conclusion and Future Work

5.1 Key Results and Conclusion

- Reverslas are studied for two Dynamo models generated by MAGNETIZER. Key differences between R19 and J24 models influence the occurrence and dynamics of reversals. A robust selection criteria was established to select evolved galaxies and detect reversals. We consider only one sub-volume for each of the model.
- The fraction of galaxies exhibiting reversals in either of the model at all redshifts is less than 5%. which is low but still higher than what current observations seem to suggest. This contrast highlights both the observational challenges involved and the need for large statistical samples to accurately determine the true occurrence and frequency of magnetic field reversals.
- Key results exhibited by the defined quantitative measure:
  - **Dimensionless Strength:** $\mathcal{S}'$ removes dependence on galaxy size and field strength.
    - Most reversals are weak: distribution of $\mathcal{S}'$ is skewed toward zero.
    - Strong reversals are rare.
  - **Asymmetry:** $\mathcal{A} = |B_2/B_1|$ quantifies field strength difference across the reversal.
    - R19: mostly $\mathcal{A} < 1$, indicating first extremum larger than second.
    - J24: peak near $\mathcal{A} \sim 1$ suggests some symmetric reversals.
  - **Radial Location:** $\mathcal{R} = r_r/r_{\max}$.
    - Distribution nearly flat in both R19 and J24; reversals can occur at any radius.
- In R19 reversals appear later, with 2 reversals (reversals) being most common. As for J24, reversals more common at early times, mostly single reversals.
- Reversal velocity and the time period over which the reversals last:
  - R19: mean $v_{\rm rev} \approx 0.4$ km/s, $\sigma \approx 0.3$ km/s. The reversals last for an average of 2.8 $Gyrs$ with $\sigma \approx 2.1$ $Gyrs$.
  - J24: mean $v_{\rm rev} \approx 0.1$ km/s, $\sigma \approx 1.7$ km/s. The reversals last for an average of $0.9 Gyrs$ with $\sigma \approx 1.6$ $Gyrs$.
- Reversal propagation:
  - Speeds generally increase with radius; outward propagation more common.
  - Pearson correlation coefficient between $v_{\rm rev}$ and radius r_r: $p \approx 0.7$.
- Merger events strongly influence reversal formation and propagation, often generating or interrupting reversals.
- We also investigated how the magnetic field of an axisymmetric galaxy with a single reversal located at half its radius would appear as observed in the Rotation Measure (RM) as function of radial distance.

5.2 Future Work

- As the physics of the work remains largely unexplored, we will focus on understanding the theory behind magnetic field reversals.
- We will perform analyses similar to those done in this project for Mr. Sukanta's fiducial model.
- When a galaxy begins evolving, its magnetic field is initialized using a seed field, and there is often a bias in the sign of the magnetic field; therefore, we will work on improving the seed field prescription.
- Following this, we will run new MAGNETIZER models with varying seed and floor fields and compare the results of these models.
- We will extend RM analysis on the MAGNETIZER simulation output to obtain RM statistics and perform mock observations to better understand the observable signatures of magnetic field reversals [**If time permits**].

6 Appendix

6.1 Derivation of the Thin–Disc Mean–Field Dynamo Equations

We will derive the Thin–Disc Mean–Field Dynamo Equations which are the governing equations solved by MAGNETIZER. We begin from the mean–field induction equation (comes directly from combining Faraday's law of induction with the MHD form of Ohm's law)

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{U} \times \mathbf{B}) + \nabla \times (\alpha \mathbf{B}) + \eta_t \nabla^2 \mathbf{B}, \tag{6.1}$$

where $\mathbf{U}$ is the mean velocity, α is the mean electromotive coefficient, and η_t is the turbulent magnetic diffusivity (assumed constant).

In cylindrical coordinates (r, φ, z) the magnetic field is

$$\mathbf{B} = (B_r,\ B_\varphi,\ B_z), \qquad \mathbf{U} = (U_r,\ U_\varphi,\ U_z).$$

The mean–field dynamo equation is

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{U} \times \mathbf{B}) + \nabla \times (\alpha \mathbf{B}) + \eta_t \nabla^2 \mathbf{B}.$$

We expand each term component–wise.

1. Expansion of $\nabla \times (\mathbf{U} \times \mathbf{B})$

First compute the cross product:

$$\mathbf{U} \times \mathbf{B} = \begin{pmatrix} U_\varphi B_z - U_z B_\varphi \\ U_z B_r - U_r B_z \\ U_r B_\varphi - U_\varphi B_r \end{pmatrix}.$$

Take its curl. The r– and φ–components are:

$$(\nabla \times (\mathbf{U} \times \mathbf{B}))_r = \frac{1}{r}\frac{\partial}{\partial \varphi}(U_r B_\varphi - U_\varphi B_r) - \frac{\partial}{\partial z}(U_z B_r - U_r B_z),$$

$$(\nabla \times (\mathbf{U} \times \mathbf{B}))_\varphi = \frac{\partial}{\partial z}(U_\varphi B_z - U_z B_\varphi) - \frac{\partial}{\partial r}(U_r B_\varphi - U_\varphi B_r).$$

2. Expansion of $\nabla \times (\alpha \mathbf{B})$

$$\alpha \mathbf{B} = (\alpha B_r,\ \alpha B_\varphi,\ \alpha B_z).$$

Hence

$$(\nabla \times (\alpha \mathbf{B}))_r = \frac{1}{r}\frac{\partial}{\partial \varphi}(\alpha B_z) - \frac{\partial}{\partial z}(\alpha B_\varphi),$$

$$(\nabla \times (\alpha \mathbf{B}))_\varphi = \frac{\partial}{\partial z}(\alpha B_r) - \frac{\partial}{\partial r}(\alpha B_z).$$

3. Expansion of $\eta_t \nabla^2 \mathbf{B}$

Using the cylindrical Laplacian

$$\nabla^2 B_i = \frac{1}{r}\frac{\partial}{\partial r}\left(r \frac{\partial B_i}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 B_i}{\partial \varphi^2} + \frac{\partial^2 B_i}{\partial z^2},$$

the diffusion terms are:

$$(\eta_t \nabla^2 \mathbf{B})_r = \eta_t \left[\frac{1}{r}\frac{\partial}{\partial r}\left(r \frac{\partial B_r}{\partial r} \right) + \frac{1}{r^2}\frac{\partial^2 B_r}{\partial \varphi^2} + \frac{\partial^2 B_r}{\partial z^2} \right],$$

$$(\eta_t \nabla^2 \mathbf{B})_\varphi = \eta_t \left[\frac{1}{r}\frac{\partial}{\partial r}\left(r \frac{\partial B_\varphi}{\partial r} \right) + \frac{1}{r^2}\frac{\partial^2 B_\varphi}{\partial \varphi^2} + \frac{\partial^2 B_\varphi}{\partial z^2} \right].$$

We adopt cylindrical coordinates (r, φ, z) and the standard thin–disc assumptions and substitue the following:

$$h \ll r, \qquad \frac{\partial}{\partial z} \gg \frac{\partial}{\partial r}, \qquad \frac{\partial}{\partial \varphi}() = 0, \qquad |B_z| \ll |B_r|, |B_\varphi|,$$

with mean rotation $\mathbf{U} = (0,\, r\Omega(r),\, U_z(z))$.

Collecting all surviving terms:

Radial component

$$\boxed{\frac{\partial B_r}{\partial t} = -\frac{\partial}{\partial z}(\alpha B_\varphi) + \eta_t \frac{\partial^2 B_r}{\partial z^2} - \frac{\partial}{\partial z}(U_z B_r)} \tag{6.2}$$

Azimuthal component

$$\boxed{\frac{\partial B_\varphi}{\partial t} = G B_r + \frac{\partial}{\partial z}(\alpha B_r) + \eta_t \frac{\partial^2 B_\varphi}{\partial z^2} - \frac{\partial}{\partial z}(U_z B_\varphi)} \tag{6.3}$$

6.2 Derivation of the Dimensionless control parameters

Introduce the dimensionless variables

$$z' = \frac{z}{h}, \qquad t' = \frac{\eta_t}{h^2}\, t, \qquad \mathbf{B} = B_0\, \mathbf{B}', \qquad \alpha(z) = \alpha_0 \sin\!\left(\pi \frac{z}{h}\right), \qquad U_z(z) = U_0 \frac{z}{h}.$$

Denote dimensionless profiles

$$\alpha'(z') = \sin(\pi z'), \qquad U_z'(z') = z'.$$

Putting these scalings into the dimensional equations

$$\frac{\partial B_r}{\partial t} = -\frac{\partial}{\partial z}(\alpha B_\varphi) + \eta_t \frac{\partial^2 B_r}{\partial z^2} - \frac{\partial}{\partial z}(U_z B_r),$$

$$\frac{\partial B_\varphi}{\partial t} = G B_r + \frac{\partial}{\partial z}(\alpha B_r) + \eta_t \frac{\partial^2 B_\varphi}{\partial z^2} - \frac{\partial}{\partial z}(U_z B_\varphi),$$

and dividing through by the chosen scales (and dropping primes on the dimensionless variables for compactness) gives the final nondimensional system

$$\frac{\partial B_r}{\partial t} = -\, R_\alpha \frac{\partial}{\partial z}\big(\alpha(z) B_\varphi\big) \;+\; \frac{\partial^2 B_r}{\partial z^2} \;-\; R_U \frac{\partial}{\partial z}\big(U_z(z) B_r\big), \tag{6.4}$$

$$\frac{\partial B_\varphi}{\partial t} = R_\omega\, B_r \;+\; R_\alpha \frac{\partial}{\partial z}\big(\alpha(z) B_r\big) \;+\; \frac{\partial^2 B_\varphi}{\partial z^2} \;-\; R_U \frac{\partial}{\partial z}\big(U_z(z) B_\varphi\big), \tag{6.5}$$

Define the standard dimensionless control parameters

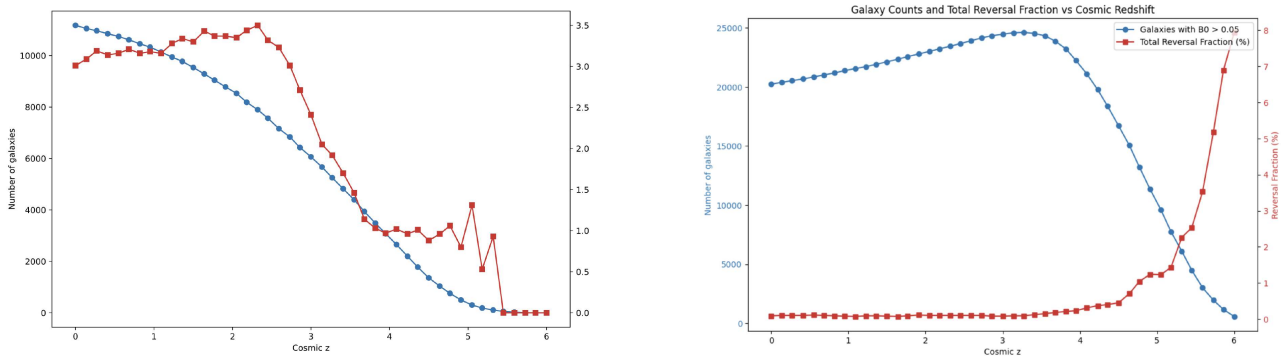
$$R_\alpha \equiv \frac{\alpha_0\, h}{\eta_t}, \qquad R_\omega \equiv \frac{G\, h^2}{\eta_t}, \qquad R_U \equiv \frac{U_0\, h}{\eta_t},$$

where $G \equiv r\,\dfrac{d\Omega}{dr}$ is the shear parameter appearing in the toroidal equation.

The equation for the floor field f can be further derived from here along with the perturbative treatment of R_U and $D \equiv R_\alpha \times R_\omega$.

6.3 Total number of galaxies that exhibit reversals

Here I want to show how many total galaxies were obtained after filtering. And how many of these galaxies exhibit atleast one reversal over redshift for both R19 and J24. This is to complement table 1.2.

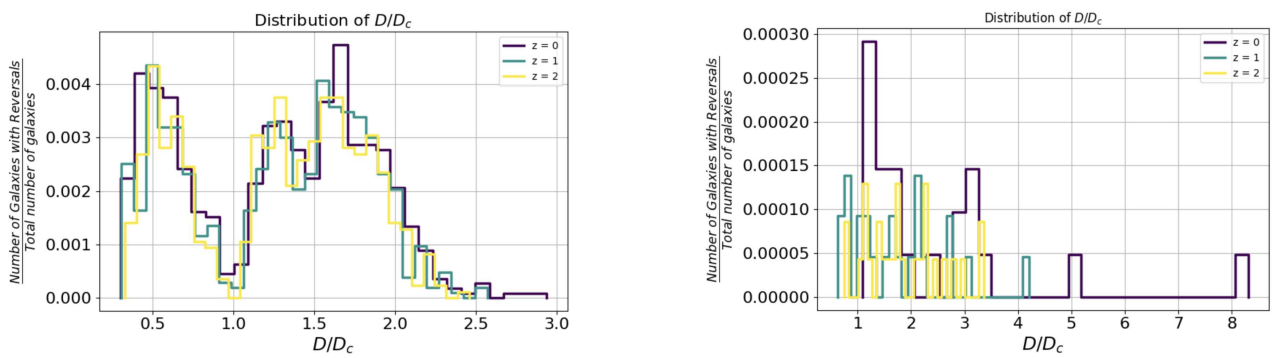


(a) R19 (b) J24

Fig. 6.1: Total number of galaxies as a function of redshift(z) (blue). Fraction of galaxies that exhibit atleast one reversal (red).

6.4 An intriguing distribution - D/D_c

Another very intriguing distribution of that I came across was for the ratio of Dynamo number over Critical Dynamo number D/D_c the galaxies which exhibit reversals.



(a) R19 (b) J24

Fig. 6.2: Distribution of D/D_c normalized to the number of galaxies exhibiting reversals to the total number of galaxies, at redshifts 0, 1 and 2.

In the case of R19 this shows up as a bimodal distribution with sharp trough at ≈ 1. While in the case of J24 it is a skewed distribution with a peak near ≈ 1.

6.5 Correlation between velocity of reversal and Halo mass

As I was dealing with correlations, in R19 I found a negative correlation between speed of the reversal and average halo mass of the galaxy but only for reversals with high speed moving outwards of the galaxy (green dots). This exhibited a person coefficient of -0.5.

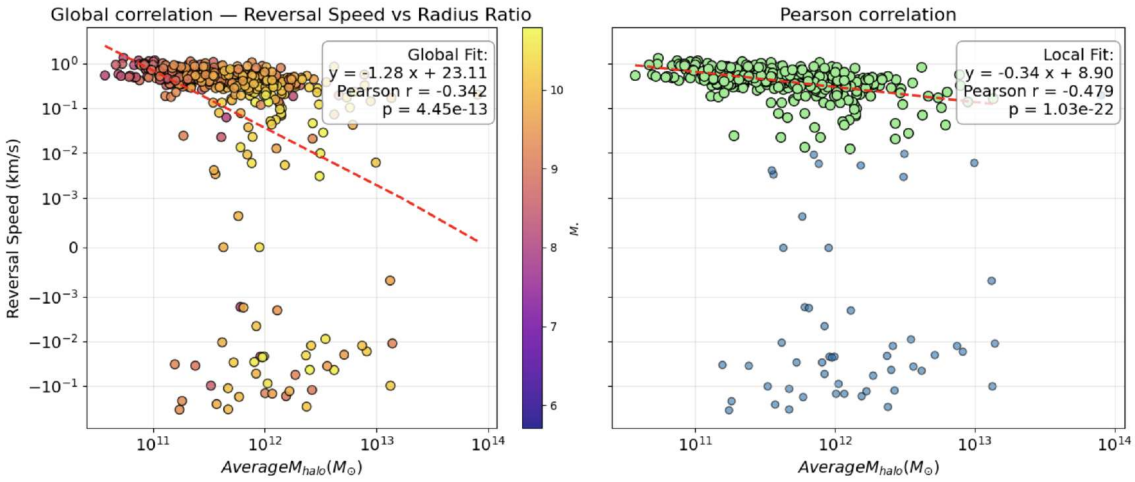


Fig. 6.3: (**Left**) Correlation of speed of the reversal (in km) against average mass of the halo. The points are coloured with respect to the average stellar mass M_* of the galaxy as indicated by the colourbar. The red line indicates a liner fit for the scatter plot. (**Right**) The green points indicate the selected points in order to calculate the Pearson coefficient ($p \approx -0.5$)

Bibliography

[1] General study of group membership.

[2] Group of galaxies within 80 mpc.

[3] Rainer Beck, Luke Chamandy, Ed Elson, and Eric G. Blackman. Synthesizing observations and theory to understand galactic magnetic fields: Progress and challenges, 3 2020. doi:10.3390/GALAXIES8010004.

[4] C. T. Becker and M. Kachelrieß. Polarized synchrotron data and the structure of the galactic magnetic field. *Physical Review D*, 111, 3 2025. doi:10.1103/PhysRevD.111.063057.

[5] Luke Chamandy, Anvar Shukurov, Kandaswamy Subramanian, and Katherine Stoker. Non-linear galactic dynamos: A toolbox. *Monthly Notices of the Royal Astronomical Society*, 443:1867–1880, 2014. doi:10.1093/mnras/stu1274.

[6] Luke Chamandy, Kandaswamy Subramanian, and Anvar Shukurov. Galactic spiral patterns and dynamo action - i. a new twist on magnetic arms. *Monthly Notices of the Royal Astronomical Society*, 428:3569–3589, 2 2013. doi:10.1093/mnras/sts297.

[7] C. L. Van Eck, J. C. Brown, J. M. Stil, K. Rae, S. A. Mao, B. M. Gaensler, A. Shukurov, A. R. Taylor, M. Haverkorn, P. P. Kronberg, and N. M. McClure-Griffiths. Modeling the magnetic field in the galactic disk using new rotation measure observations from the very large array. *Astrophysical Journal*, 728, 2 2011. doi:10.1088/0004-637X/728/2/97.

[8] P Frick, R Stepanov, A Shukurov, and D Sokoloff. Structures in the rotation measure sky.

[9] J L Han, R N Manchester, A G Lyne, G J Qiao, and W Van Straten. Pulsar rotation measures and the large-scale structure of galactic magnetic field, 2006.

[10] Charles Jose, Luke Chamandy, Anvar Shukurov, Kandaswamy Subramanian, Luiz Felippe S. Rodrigues, and Carlton M. Baugh. Understanding the radio luminosity function of star-forming galaxies and its cosmological evolution. *Monthly Notices of the Royal Astronomical Society*, 532:1504–1521, 8 2024. doi:10.1093/mnras/stae1426.

[11] Kohei Kurahara and Hiroyuki Nakanishi. Magnetic field vector structure of ngc6946. *Galaxies*, 7, 2019. doi:10.3390/GALAXIES7020059.

[12] Y. K. Ma, S. A. Mao, A. Ordog, and J. C. Brown. The complex large-scale magnetic fields in the first galactic quadrant as revealed by the faraday depth profile disparity. *Monthly Notices of the Royal Astronomical Society*, 497:3097–3117, 9 2020. doi:10.1093/mnras/staa2105.

[13] Evgeny Mikhailov and Tatiana Khasaeva. Evolution of the magnetic field reversals in galaxies.

[14] D. Moss, R. Stepanov, T. G. Arshakian, R. Beck, M. Krause, and D. Sokoloff. Multiscale magnetic fields in spiral galaxies: Evolution and reversals. *Astronomy and Astrophysics*, 537, 2012. doi:10.1051/0004-6361/201118122.

[15] T. Nota and P. Katgert. The large-scale magnetic field in the fourth galactic quadrant. *Astronomy and Astrophysics*, 513, 4 2010. doi:10.1051/0004-6361/200911803.

[16] Rüdiger Pakmor, Rebekka Bieri, Freeke van de Voort, Maria Werhahn, Azadeh Fattahi, Thomas Guillet, Christoph Pfrommer, Volker Springel, and Rosie Y. Talbot. Magnetic field amplification in cosmological zoom simulations from dwarf galaxies to galaxy groups. *Monthly Notices of the Royal Astronomical Society*, 528:2308–2325, 2 2024. doi:10.1093/mnras/stae112.

[17] V. Pelgrims, J. F. MacIás-Pérez, and F. Ruppin. Galactic magnetic field reconstruction using the polarized diffuse galactic emission: Formalism and application to planck data. *Astronomy and Astrophysics*, 652, 8 2021. doi:10.1051/0004-6361/201833962.

[18] L. F.S. Rodrigues, L. Chamandy, A. Shukurov, C. M. Baugh, and A. R. Taylor. Evolution of galactic magnetic fields. *Monthly Notices of the Royal Astronomical Society*, 483:2424–2440, 2 2019. doi:10.1093/mnras/sty3270.

[19] B. Ruiz-Granados, J. A. Rubiño-Martín, and E. Battaner. Constraining the regular galactic magnetic field with the 5-year wmap polarization measurements at 22 ghz. *Astronomy and Astrophysics*, 522, 11 2010. doi:10.1051/0004-6361/200912733.

[20] Y. Stein, R. J. Dettmar, J. Irwin, R. Beck, M. Wezgowiec, A. Miskolczi, M. Krause, V. Heesen, T. Wiegert, G. Heald, R. A.M. Walterbos, J. T. Li, and M. Soida. Chang-es: Xiii. transport processes and the magnetic fields of ngc 4666: Indication of a reversing disk magnetic field. *Astronomy and Astrophysics*, 623, 3 2019. doi:10.1051/0004-6361/201834515.

[21] X. H. Sun, W. Reich, A. Waelkens, and T. A. Enßlin. Radio observational constraints on galactic 3d-emission models. *Astronomy and Astrophysics*, 477:573–592, 1 2008. doi:10.1051/0004-6361:20078671.

[22] Michael Unger and Glennys R. Farrar. The coherent magnetic field of the milky way. *The Astrophysical Journal*, 970:95, 7 2024. doi:10.3847/1538-4357/ad4a54.

[23] Fabian Walter, Michael Dahlem, and Ute Lisenfeld. The superwind galaxy ngc 4666: Gravitational interactions and the influence of the resulting starburst on the interstellar medium. URL: http://www.iram.es/IRAMES/telescope/telescopeSummary/telescope.

STATISTICAL PROPERTIES OF RM

1 RM OF SINGLE GALAXY

1.1 PDF of $B_{||}$

To obtain the PDF of RM of a single galaxy, we start by looking at the PDF of $B_{||}$ (Eq. 1).

$$B_{||} = -B_0\ e^{-r/r_B}\ cos(\theta - p)\ sini \ ...(1)$$

Where $r_B = 20kpc$.

We consider the uniform distribution of θ (Eq. 2). We have used the formalism mentioned in Appendix A. Then, I used rejection sampling to find the PDF of trigonometric functions, such as $cos(\theta - p)$, in this case. In this section, the angle of inclination is set to 30⁰.

$$f(\theta) = \begin{cases} \frac{1}{2\pi}, & 0 \leq \theta \leq 2\pi \\ 0, & \text{otherwise} \end{cases} \ ...(2)$$

We can consider two possible distributions for r as indicated in equations 4 and 5. We then find the PDF of $B_0\ e^{r/r_B}$ (similar to the trigonometric functions).

$$f_R(r) = \frac{6}{R_{\max}^{\eta+1}\Gamma(\frac{\eta+1}{6})}\ r^{\eta} e^{-\left(\frac{r}{R_{\max}}\right)^6}, \quad 0 < r < \infty \ ...(4)$$

$$f_R(r) = \frac{\eta+1}{R_{\max}^{\eta+1}} r^{\eta}, \quad 0 < r < \infty \ ...(5)$$

Using brute force, we find the PDF of $B_{||}$ considering the uniform distribution of θ and using the distribution for r as in equation 4 to be,

$$f_{B_{||}}(b_{||}) = \begin{cases} \frac{k_1}{|b_{||}|}[\arcsin(\frac{|b_{||}|}{a}) - \arcsin(\frac{|b_{||}|}{b})], & -a \leq b_{||} \leq a, \\ \frac{k_1}{|b_{||}|}\arccos(\frac{|b_{||}|}{b}), & b_{||} \in [-b, -a) \cup (a, b]\,. \end{cases} \ ...(6)$$

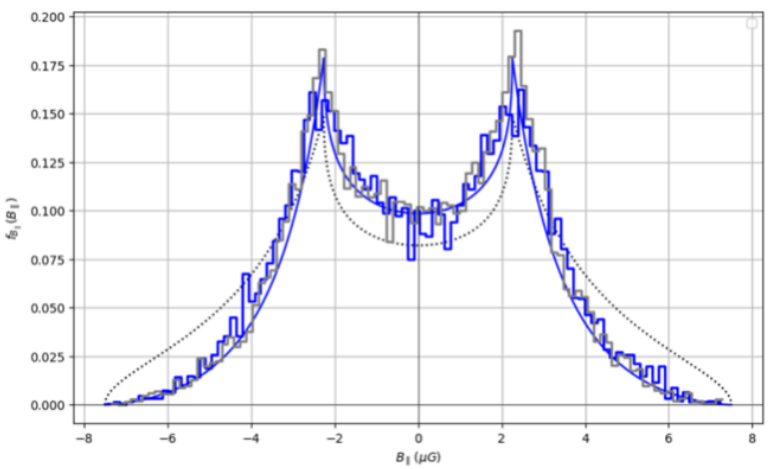
Where $k_1 = r_B/[\pi(R_{max} - R_{min})]$ & $a = B_0 \sin i e^{-R_{max}/r_B}$ and $b = B_0 \sin i e^{-R_{min}/r_B}$.
For the PDF of $B_{||}$ considering the uniform distribution of θ and using the distribution for r as in equation 5 with $\eta = 1$ to be,

$$f_{B_\parallel}(b_\parallel) = \begin{cases} \frac{k_2}{|b_\parallel|}\frac{R_{max}}{r_B}[\arcsin(\frac{|b_\parallel|}{a}) - \arcsin(\frac{|b_\parallel|}{b})], & -a \le b_\parallel \le a, \\ \frac{k_2}{|b_\parallel|}\ln(\frac{B_0 sini}{|b_\parallel|})[\arccos(\frac{|b_\parallel|}{b})], & b_\parallel \in [-b,-a) \cup (a,b]\,. \end{cases} \quad ...(7)$$

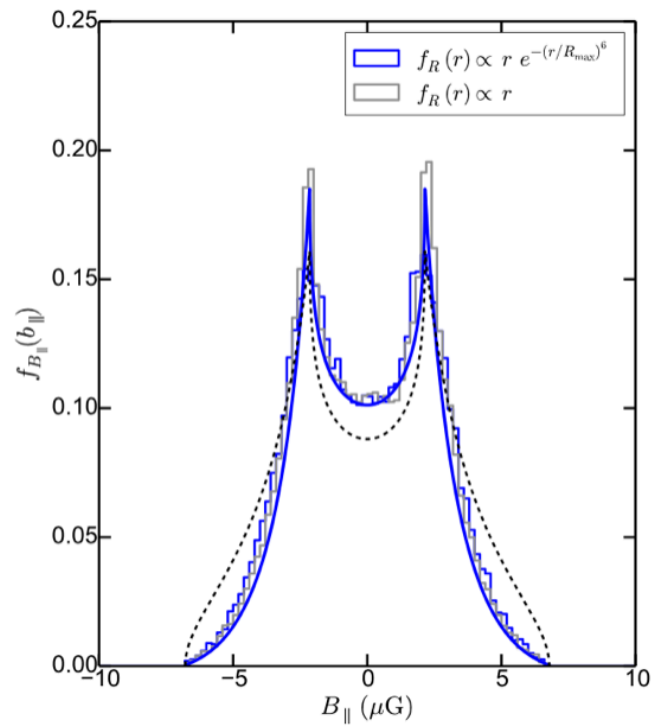
Where $k_2 = r_B/(\pi R_{max})$ & a and b are the same as before.

NOTE: The histogram-type plots indicate the results from sampling, and the solid lines are analytical.

In the first figure, we try to look at the difference between PDF of $B_\parallel$ for different PDFs of r as given in equations 4 and 5. Here, $B_0 = 15\mu G$, $R_{min} = 0\ kpc$, $R_{max} = 24\ kpc$ and $\eta = 1$.



(a)

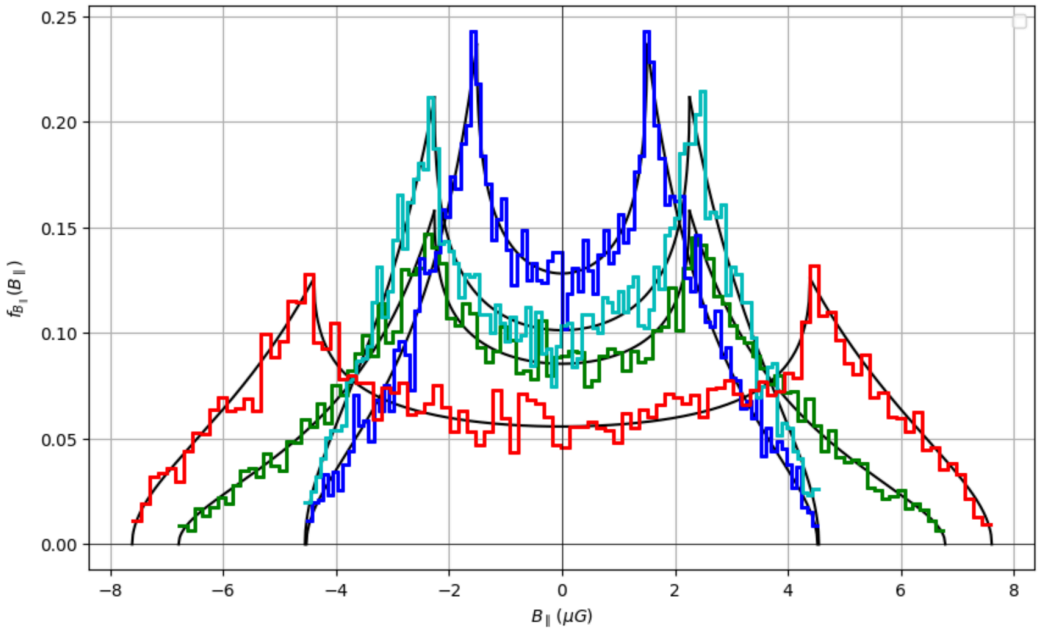


(b)

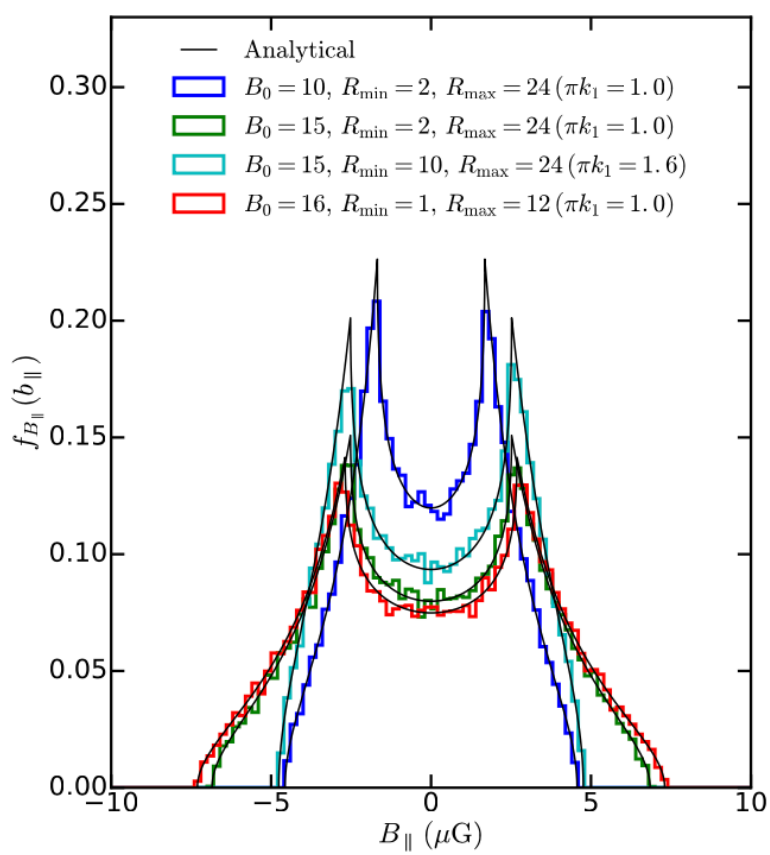
Figure 1: *(a)* Reproduced Result, *(b)* Original. Distribution of B∥ for the case when radii are distributed as per equation (4) with η = 1 is shown as the blue histogram. The blue curve is the approximated analytical PDF given in equation (7). B∥ for uniform distribution of radii with the same parameters is shown as the dashed line. The grey histogram is the distribution of B∥ for the case when the distribution of impact radii given by equation (4) is approximated as equation (5).

The original and the reproduced results are in agreement with each other.

Now, we consider a uniform distribution for r as given in equation 8 (here we have used inverse sampling). We consider different values of B_0, R_{min} and R_{min} and plot PDF of $B_{\|}$.



(a)



(b)

Figure 2: *(a)* Reproduced Result *(b)* Original $B_{\|}$ Distribution. Distribution of $B_{\|}$ for a galaxy inclined at *i*= 30⁰ for uniform distribution of radius. The black lines are the analytical PDF of RM as given in equation (6).

$$f(\theta) = \begin{cases} \frac{1}{R_{max} - R_{min}}, & R_{min} \leq r \leq R_{max} \\ 0, & \text{otherwise} \end{cases} \quad ...(8)$$

All the distributions are in agreement with each other except for the parameters $B_0 = 16\mu G$, $R_{min} = 1\ kpc$ and $R_{max} = 12\ kpc$ (red). In order to check this case, we check the value of the cusps which occur at $-a$ and $+a$.

$$a = B_0 \sin i e^{-R_{max}/r_B} = 16\mu G \times sin(30^0) \times e^{-12kpc/20kpc} \approx 4.4\mu G$$

Which shows an agreement with the reproduced result (Fig. 2 (a)).

1.2 PDF of RM

The expression for RM is given by

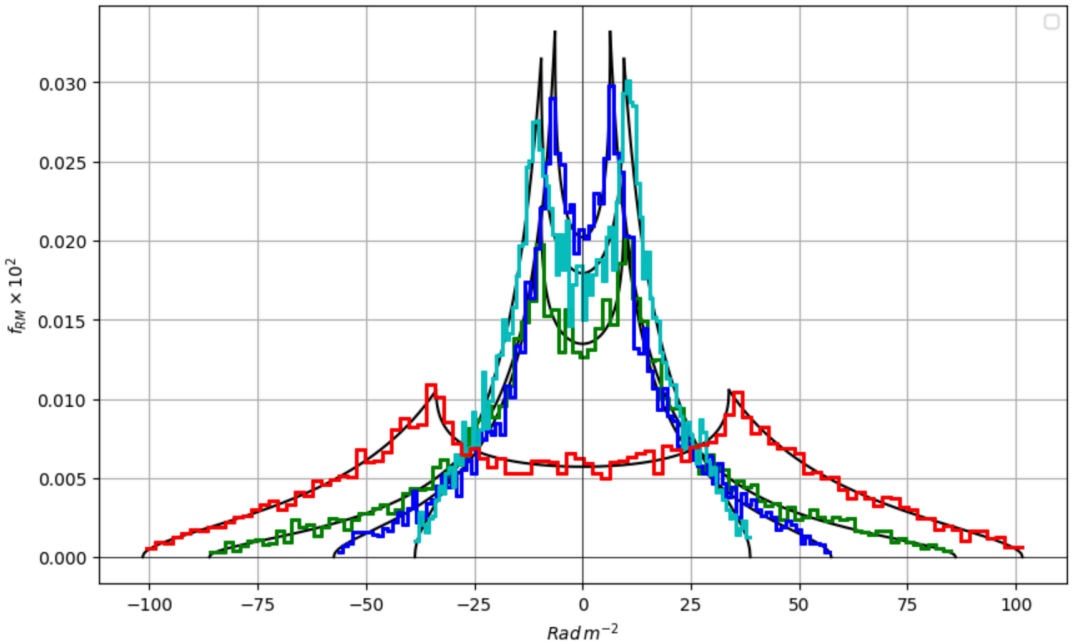
$$RM = -0.81\ B_0\ n_0 e^{-r/r_0'}\ h_{ion} \tan i\ \ \cos(\theta - p) \quad ...(9)$$

Here r and θ have a uniform distribution. The values of $h_{ion} = 500kpc$, $n_0 = 0.03\ cm^{-1}$, $r_0 = 10\ kpc$ and $i = 30^0$. We consider different values of B_0, R_{min} and R_{min} and plot PDF of RM.

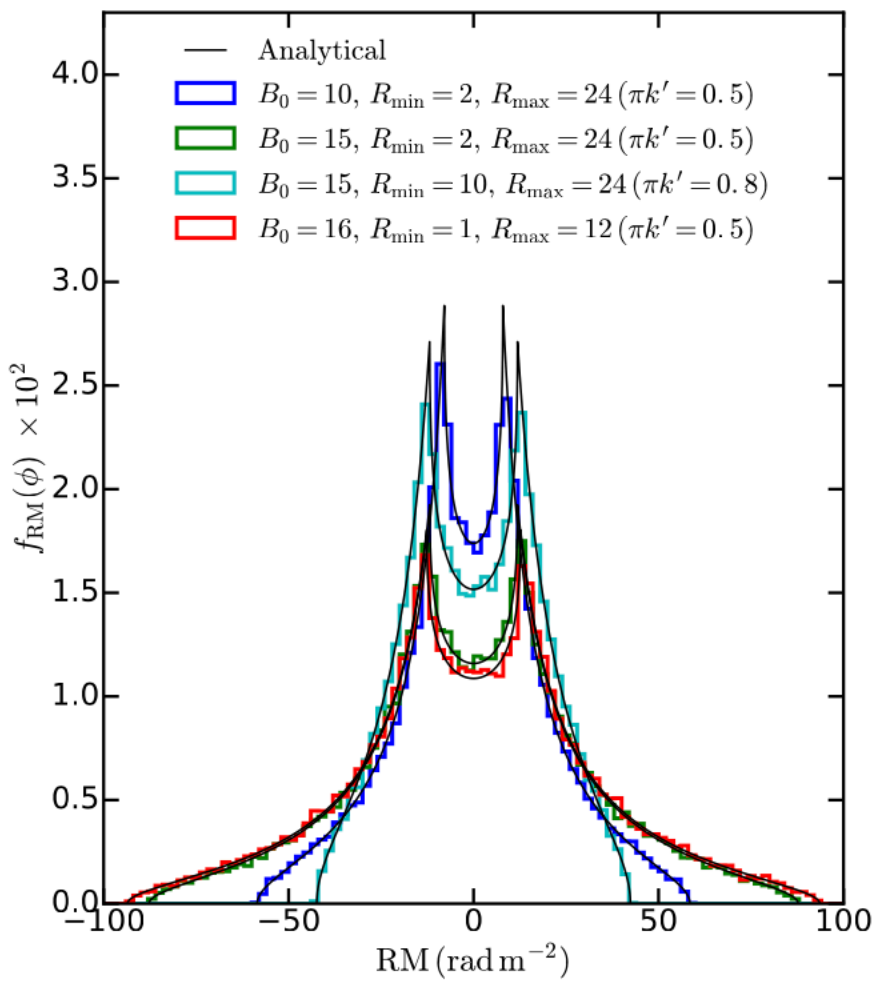
Using brute force, we find the PDF of RM for the same parameters to be,

$$f_{RM}(\phi) = \begin{cases} \frac{k'}{|\phi|}[\arcsin(\frac{|\phi|}{a}) - \arcsin(\frac{|\phi|}{b})], & -a \leq \phi \leq a, \\ \frac{k'}{|\phi|} \arccos(\frac{|\phi|}{b}), & \phi \in [-b, -a) \cup (a, b]\,. \end{cases} \quad ...(10)$$

Where $k' = r_0'/[\pi(R_{max} - R_{min})]$,$a' = 0.81 B_0 n_0 h_{ion} \tan i e^{-R_{max}/r_0'}$ and $b' = 0.81 B_0 n_0 h_{ion} \tan i e^{-R_{min}/r_0'}$.



(a)



(b)

Figure 3: *(a)* Reproduced Result *(b)* Original RM Distribution. Distribution of RM for a galaxy inclined at i= 30⁰ for uniform distribution of radius. The black lines are the analytical PDF of RM as given in equation (10).

The disagreement is again for the same set of parameters as for $B_{||}$. All the distributions agree with each other except for the parameters $B_0 = 16\mu G$, $R_{min} = 1\ kpc$ and $R_{max} = 12\ kpc$ (red). To check this case, we check the value of the cusps which occur at $-a'$ and $+a'$.

$$a' = 0.81 B_0 n_0 h_{ion} \tan i e^{-R_{max}/r_0'} = 0.81 \times 16\mu G \times 0.03 cm^{-3} \times 500 kpc \times \tan(30^0) \times e^{-12kpc/10kpc}$$
$$\approx 34\ rad\ m^{-2}$$

This is in agreement with the reproduced result (Fig. 3 (a))

2 REFERENCE

[1] Basu, A., Mao, S. A., Fletcher, A., Kanekar, N., Shukurov, A., Schnitzeler, D., Vacca, V., & Junklewitz, H. (2018). Statistical properties of Faraday rotation measure in external galaxies – I. Intervening disc galaxies. *Monthly Notices of the Royal Astronomical Society*, *477*(2), 2528–2546. https://doi.org/10.1093/mnras/sty766

[1] Please find the code and the results on my GitHub page,
LINK: https://github.com/HShishir1729/GALAXY.git

STATISTICAL PROPERTIES OF RM

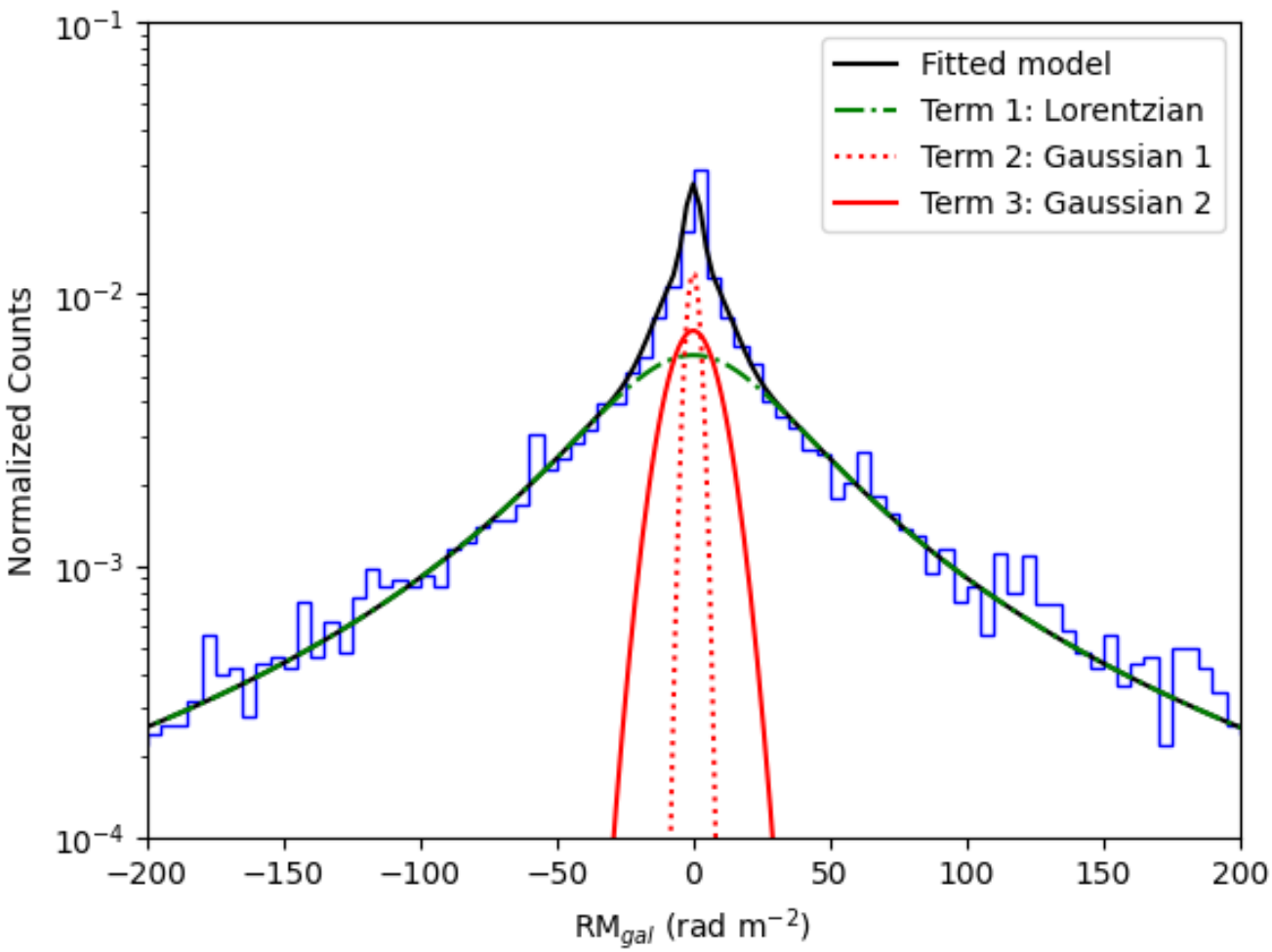
1 RM FOR SAMPLE OF GALAXIES

For the case of a sample of galaxies, as suggested, I have considered a uniform distribution for $r,\ \theta,\ and\ \phi$. And a Gaussian distribution for B_0 and n_e.

$$P(B_0) = \frac{1}{\sqrt{2\pi\sigma_{B_0}^2}} \exp\left(-\frac{(B_0 - \langle B_0 \rangle)^2}{2\sigma_{B_0}^2}\right) \ldots 1$$

$$P(n_e) = \frac{1}{\sqrt{2\pi\sigma_{n_e}^2}} \exp\left(-\frac{(n_e - \langle n_e \rangle)^2}{2\sigma_{n_e}^2}\right) \ldots 2$$

I performed sampling for 10000 galaxies and obtained RM_{gal}.



For the figure given, the values of the parameters that I have used are:

-> $< B_0 > = 15\mu B$
-> $\sigma_{B_0} = 5\mu B$
-> $< n_e > = 0.03cm^{-3}$
-> $\sigma_{n_e} = 0.01cm^{-3}$

Also, in the case of uniform distribution of r, the r_{min} and r_{max} values are 2kpc and 25kpc, respectively.

I have tried to fit the data to the following functional form of RM_{gal} (using the `curve_fit` function of `scipy.optimize`) as indicated in the paper. The fit gives the parameters as the output.

$$f_{\text{RM}_{\text{gal}}}(x) = a_1[\frac{w_1^2}{w_1^2 + (x - m_1)^2}] + a_2 \exp\left(-\frac{(x - m_2)^2}{w_2}\right) + a_3 \exp\left(-\frac{(x - m_3)^2}{w_3}\right) \ldots 3$$

```
a1 = 0.0060
m1 = 0.0023
w1 = 42.2485
a2 = 0.0121
m2 = -0.0009
w2 = 14.1187
a3 = 0.0073
m3 = 0.0009
w3 = 197.0940
```

Here, a_1, a_2 and a_3 are aplitudes normalizations w_1, m_1, m_2 and m_3 have the units of rad m^{-2} and w_2 and w_3 have the units (rad m^{-2})$^{-2}$.

2 REFERENCE

[1] Basu, A., Mao, S. A., Fletcher, A., Kanekar, N., Shukurov, A., Schnitzeler, D., Vacca, V., & Junklewitz, H. (2018). Statistical properties of Faraday rotation measure in external galaxies– I. Intervening disc galaxies. Monthly Notices of the Royal Astronomical Society, 477(2), 2528–2546. https://doi.org/10.1093/mnras/sty766

[2] Please find the code and the results on my GitHub page,
LINK: https://github.com/HShishir1729/GALAXY.git (RM_10000_CLEAN)

STATISTICAL PROPERTIES OF RM

1 RM FOR SAMPLE OF GALAXIES

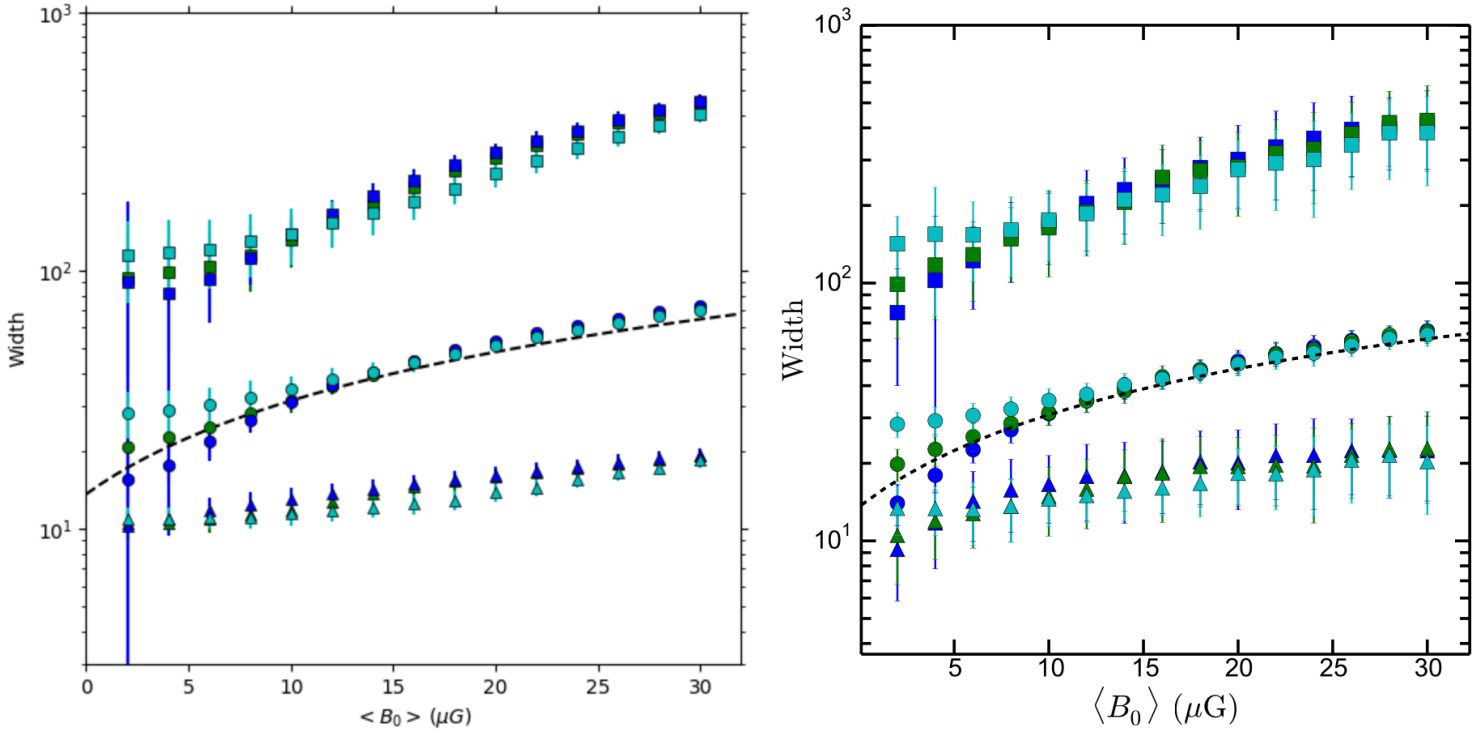


FIG. 1: a) My result and **b)** Result from the paper. Variation of the widths of the modelled components with $\langle B_0 \rangle$. Circles, triangles, and squares are for w_1, w_2, and w_3, the widths of the Lorentzian and the two Gaussian components, respectively. The different colours are for different values of σ_{B0} (blue = 2 μG, green = 5 μG, cyan =10 μG). The dashed line is the empirical fit to w1 as a function of $\langle B_0 \rangle$ given by equation (1).[1]

The dashed line is fit according to the model by the formula,

$$w_1(< B_0 >) = p_0 + p_1 < B_0 > + p_2 < B_0 >^2 \quad ...1$$

The values that I obtained for the fit are 13.6 rad m^{-2}, 1.8 (rad $m^{-2})^2$, -0.0032 (rad $m^{-2})^2$. The values in the paper were 13.6 rad m^{-2}, 1.8 (rad $m^{-2})^2$, -0.0076 (rad $m^{-2})^2$.

2 REFERENCE

[1] Basu, A., Mao, S. A., Fletcher, A., Kanekar, N., Shukurov, A., Schnitzeler, D., Vacca, V., & Junklewitz, H. (2018). Statistical properties of Faraday rotation measure in external galaxies– I. Intervening disc galaxies. Monthly Notices of the Royal Astronomical Society, 477(2), 2528–2546. https://doi.org/10.1093/mnras/sty766

[2] Please find the code and the results on my GitHub page,
LINK: https://github.com/HShishir1729/GALAXY.git (RM_10000_CLEAN(2))

CONTENT

SL. NO.	HEADING	PAGE NO.
1	APSERa SIMULATION PIPELINE FOR CROSS ORTHOGONAL POLARIZATION ORIGINAL CODE	2
1.1	GENERATING A SKY MODEL	2
1.2	REPLACE BEAM WITH ANALYTIC BEAM	3
1.3	GETTING RECOMBINATION LINES	9
1.4	GETTING SPECTRA	12
2	APSERa SIMULATION PIPELINE FOR CROSS ORTHOGONAL POLARIZATION UPDATED CODE	18
2.1	STOKES PARAMETERS	19
2.2	SIMULATION OF GLOBAL SIGNAL OBSERVATIONS	21
2.3	DOUBLE REFRACTION AND RETARDER PLATES	29
2.4	ROTATING QUARTER WAVE PLATE	38
3	MS FUNCTION	43
3.1	SIGNAL EXTRACTION	43
3.2	FITTING RESIDUALS	44
3.3	BAYESIAN INFERENCE	51
3.4	APPLYING BAYESIAN INFERENCE ON HALF-WAVE / QUARTER-WAVE PLATE	56
4	LINKS	59
5	JUPYTER NOTEBOOK CONTENT	60

1. APSERa Simulation Pipeline for Cross Orthogonal Polarization Original Code:

1.1. Generating a Sky Model:

We start by downloading and accessing the necessary files.

Ensure that 3 file paths are set correctly for your machine before running

1. beam_path: This contains the beam files
2. PATH: this contains the path to the fitting functions
3. datafile_path: This is where the output residual after fitting will be saved
4. sky_map_path: Where the sky maps are located
5. recomb_path: Where the recombination line data is located

```
#beam_path = "/home/linux2/Downloads/Farfield/farfield_finite/140mm_ground"
beam_path = "/home/mayuris/workspace/RRI_DISTORTION_Lab/APSERa/Keerthi_thesis/Data/Antenna_meas
    /140mm_ground"

#PATH = '/home/linux2/Downloads/msfitcodes/'
PATH = '/home/mayuris/workspace/saras3/'


#datafile_path = "/home/linux2/Downloads/Farfield/farfield_finite/140mm_ground/residues_final.txt
    "
datafile_path = "/home/mayuris/workspace/RRI_DISTORTION_Lab/APSERa/Keerthi_thesis/residues_final.
    txt"

#sky_map_path = "."
sky_map_path = "/home/mayuris/workspace/RRI_DISTORTION_Lab/APSERa/Keerthi_thesis/Sky_model"

#recomb_path = "."
recomb_path = "/home/mayuris/workspace/RRI_DISTORTION_Lab/APSERa/Theory"
```

Now that our files have been located we import the necessary packages.

```
# For computation and mathematics
import numpy as np
import math
import ephem
from scipy.signal import correlate
from scipy.optimize import curve_fit
from scipy.interpolate import Rbf, InterpolatedUnivariateSpline, RegularGridInterpolator

#For working with sky coordinates and similar such arrangements
import healpy as hp

#For plotting
import matplotlib.pyplot as plt
%matplotlib inline

#For generating sky model
import pysm3
import pysm3.units as u
from pysm3 import CMBMap

#Astronomical conversions
from astropy.timeseries import TimeSeries, BinnedTimeSeries
from astropy.time import Time, TimeDelta
from astropy.coordinates import Galactic, SkyCoord, FK5, EarthLocation, AltAz, Angle
import astropy.units as u

#For working with data sets
import pandas as pd

#For CMB emission
import camb
from camb import model, initialpower
```

We now get the time and the number of spectra.

- start-time-in-UTC –> A string representing the current date and time in ISO format (e.g., '2023-10- 03T12:34:56.789'). It's obtained using the datetime.now() function.
- n-spectra –> The number of spectra to be taken, can range from 1 - infinity.
- delta-t-in-hours –> The time interval between each spectrum in hours, set to 1 hour in this code.
- t –> An instance of the Time class created from start-time-in-UTC. It represents a specific point in time and is used for time-related calculations.
- ts –> A TimeSeries object representing a sequence of time instances. It starts from start-time-in-UTC and advances by delta-t-in-hours hours for each sample. The ts object contains the time instances for the spectra.

```

#Getting time and number of spectra

from datetime import datetime   # importing datetime module

start_time_in_UTC = str(datetime.now())
#print(start_time_in_UTC)

n_spectra = 1              #Number of spectra to be taken
delta_t_in_hours = 1       #Time interval between each spectra needed in hours

t = Time(start_time_in_UTC, format = 'iso', scale= 'utc')

ts = TimeSeries(time_start= start_time_in_UTC, time_delta= delta_t_in_hours *u.hour, n_samples=
    n_spectra)
ts = ts.time
print(ts)

['2023-10-09 10:54:33.064']
```

1.2. Replace beam with analytical beam:

The initial part of the code helps read the beam file and…

In the second part of the code, the beam function of the antenna is a function of the frequency, the zenith angle and the azimuth angle. That is, $f(\nu, \theta, \phi)$. Here we consider a toy model of the antenna beam pattern wherein, the ideal beam $f(\nu, \theta, \phi) = sin^2(\theta)$ is considered, for $0^0 < \theta < 90^0$ and $f(\nu, \theta, \phi) = 0$, for $90^0 < \theta < 180^0$. We notice that the beam pattern gives a certain weight to the incoming signal. For example, a signal detected at an angle of $\theta = 0^0$ the signal will be given a weight of "1" and a signal detected at an angle of $\theta = 45^0$ the signal will be given a weight of "1/2" and so on.

```python
import glob
import os
import numpy as np
from scipy import interpolate

'''
## In the rt_file, mention the filename of S11. Please check if it is in linear form.

rt_file    = 'dipole_correct.txt'
file_list  = sorted(glob.glob(os.path.join(beam_path,"*farfield*.txt")))
#print(file_list)



gamme_file = os.path.join(beam_path, rt_file)

gamma_freq = []
gamma_val  = []

with open(gamme_file) as fa:
    for line_aa in fa.readlines()[2:]:
        line_aa = line_aa.strip()
        col1    = line_aa.split('\t')
        _freq   = np.array(list(map(float, col1[0].split())))[0]
        _val    = np.array(list(map(float, col1[1].split())))[0]
        gamma_freq.append(_freq)
        gamma_val.append(_val)

gamma_freq = np.array(gamma_freq)
gamma_val  = np.array(gamma_val)
gamma_val=10**(gamma_val/20)

print(gamma_freq)
print(gamma_val)
gamma_func = interpolate.interp1d(gamma_freq, gamma_val)  '''


## This below print statement is to verify our interpolation. print(gamma_func(any freq value))
## and check if it matches the value in the file
#print(gamma_func(2.8))


## The below code is for the beam definition. Please make sure to check the beam_def function before proceeding

'''
phi_res   = 1
theta_res = 1

phi_array   = np.arange(0, 360, phi_res)
theta_array = np.arange(90, -90-theta_res, -theta_res)
freq_array  = []
file_array  = []

def get_freq_from_file(filename):
    _temp = os.path.basename(filename).replace('.txt','').replace('farfield','').replace('(f=','').replace(')','').replace('[1]','')
    return float(_temp)
```

```python

for ii, file_add in enumerate(file_list):
    freq_array.append(get_freq_from_file(file_add))
    file_array.append(file_add)

print(get_freq_from_file('farfield(f=2.608)[1].txt'))

np.testing.assert_array_equal(file_array, file_list)

freq_array = np.array(freq_array)
file_array = np.array(file_array)

isort      = np.argsort(freq_array)
freq_array = freq_array[isort]
file_array = file_array[isort]

beam_3D_unnorm = np.zeros((len(freq_array), len(theta_array), len(phi_array)))

'''
'''
for ii, freq in enumerate(freq_array):
    print("Processing frequency {} GHz".format(freq))
    file_add = file_array[ii]
    with open(file_add) as fa:
        for line_aa in fa.readlines()[2:]:
            line_aa = line_aa.strip()
            col1    = line_aa.split('\t')
            all_val = np.array(list(map(float, col1[0].split())))

            theta   = 90-all_val[0]
            phi     = all_val[1]
            beam    = all_val[2]
            print(theta)


            iphi    = np.where(phi_array==phi)[0][0]
            itheta  = np.where(theta_array==theta)[0][0]
            beam_3D_unnorm[ii, itheta, iphi] = 10**(beam/10)



beam_3D = np.zeros((len(freq_array), len(theta_array), len(phi_array)))

print(beam_3D_unnorm[0, :,:].shape)


for ifreq in range(beam_3D.shape[0]):
    for itheta in range(beam_3D.shape[1]):
        for iphi in range(beam_3D.shape[2]):
            beam_3D[ifreq, itheta,iphi] = beam_3D_unnorm[ifreq, itheta,iphi]/np.amax(beam_3D_unnorm[ifreq,:,:])


print(freq_array)


theta_array_new = np.flip(theta_array)
phi_array_new   = phi_array
beam_3D_flip=np.flip(beam_3D,axis=1)
print(beam_3D_flip[2,:,0],beam_3D[2,:,0])
from scipy.interpolate import RegularGridInterpolator
```

```
my_interpolating_function = RegularGridInterpolator((freq_array, theta_array_new,
                                                     phi_array_new), beam_3D_flip)
print(my_interpolating_function.shape)

'''

## Beam definition
## If only the effect of S11 has to be checked (with ideal antenna beam), use the sin^(theta)
    beam.

## Added

#freq = [408, 1420, 23000]
#phi_res   = 1
#theta_res = 1

#az   = np.arange(0, 360, phi_res)
#alt = np.arange(90, -90-theta_res, -theta_res)

## Added ends

def beam_def(freq, alt, az):
    if (alt<=0):
        return 0
    else:
        return (math.sin(alt*PI/180)*math.sin(alt*PI/180))



#plt.plot(gamma_freq, gamma_val)
#plt.xlim([2,4])
#plt.show()
```

We will define all the constants and set the dimensions of the variables being used.

- P_hot (1001,) --> This represents the power or intensity associated with a "hot" load or source. In the context of calibration, a hot load is a reference source with a known temperature (in this case, represented by T_hot) used to calibrate the receiver system. The formula used to calculate P_hot is derived from Planck's radiation law.
- P_cold (1001,) --> This variable represents the power or intensity associated with a "cold" load or source. Similar to P_hot, a cold load is another reference source with a known temperature (in this case, represented by T_cold) used for calibration.
- P_dif (1001,) --> The subtraction of the power associated with the hot load from the power associated with the cold load.

The expression for power associated with a load is given by, $P = \dfrac{2h\nu}{e^{\frac{h\nu}{kT}} - 1} \ldots 1$

- cross_mix --> Is used to store cross-polarization terms or mixed terms in signal processing.
- spectra_real --> This array stores the real part of the spectra at each time step.
- spectra_imag --> This array stores the imaginary part of the spectra at each time step.

```python

#Basic Definitions - all of them being in SI Units

h = 6.62607157e-34                          # Plank's Constant
K_Bolt = 1.3806488e-23                      # Boltzmann Constant
c = 2.99792458e+08                          # Speed of light
TCMB = 2.72548                              # CMB Temperature
f_start = 2                                 # Check the f_start and f_stop and change the number of channels accordingly. Can be used to define frequency range.
f_stop = 4                                  # Frequency range end point
No_of_Channels = 1001                       # no.of frequency channels i.e. number of frequencies into consideration
Channel_width = 0.01
nside = 16                                  # Parameter which determines the resolution of HealPy map.
T_hot = 3730                                # Hot temerature threshold for absolute calibration - in Kelvin units.
T_cold = 2730                               # Cold temerature threshold for absolute calibration - in Kelvin units.
Time_in_UTC = ts                            # It represents the time in Coordinated Universal Time (UTC) when the spectra or observations are taken
latitude = 13.067439                        # Longitude in degrees
longitude =80.237617                        # Latitude in degrees
altitude = 0                                # Altitude in meters
alpha = 0.0065                              # Temp. lapse rate (deg C per meter) dry adiabatic lapse rate" (DALR), and it describes how the temperature of dry air changes with altitude as it rises in the Earth's atmosphere.
PI = 3.14159265                             # Pi value
Noise_int_time = 3e+13                      # It specifies the integration time for noise calculations in the systematics.
Trx = 14                                    # It represents the receiver noise temperature in Kelvin.
T_atm = 1                                   # It represents the atmospheric noise temperature in Kelvin.
num_components = 3                          #no. of components in the sky emission i.e. I, Q & U


# These variables are used to define the dimensions


frequencies = np.linspace(f_start,f_stop, No_of_Channels, endpoint=True)
CMB_Int = np.empty([No_of_Channels])
P_hot = np.empty(No_of_Channels)
P_cold = np.empty(No_of_Channels)
P_dif = np.empty(No_of_Channels)
cross_real = [0]*No_of_Channels
cross_imag = [0]*No_of_Channels
sumwt = [0]*No_of_Channels
sigma = [0]*No_of_Channels
spectra = [0]*n_spectra
spectra_cross_imag = [0]*n_spectra
#NPIX = hp.nside2npix(N_SIDE)

cross_mix = [0]*No_of_Channels
spectra_real = [0] * n_spectra
spectra_imag = [0] * n_spectra
sigma_real = [0]*No_of_Channels
sigma_imag = [0]*No_of_Channels
```

- sigma_real --> This array stores the real part of the noise (sigma) added to the spectra at each channel. The noise is calculated and added to the spectra to simulate realistic observations.
- sigma_imag --> This array stores the imaginary part of the noise (sigma) added to the spectra at each channel.

To correct for the incoming EM wave due to the atmospheric refractive index we use the Barometric Equation given by $P_h = P_0 e^{(\frac{-mgh}{kT})}$, where $'P_0'$ is the pressure on the ground, 'm' is the

mass of one molecule, 'h' is the height from the surface and 'k' is the Boltzmann's constant. And the formula used for, R, $R = \frac{1.02}{tan((\theta + \frac{10.3}{\theta + 5.11}) \times \frac{\pi}{180})}$.

The code defines a Python function refraction(alt_in_deg, altitude) that calculates the apparent altitude of an object in the sky after accounting for atmospheric refraction.

alt_in_deg --> This parameter represents the true altitude of the object in degrees.
altitude --> This parameter represents the observer's altitude above sea level in meters.

The function performs the following steps:

- pres --> Calculates the atmospheric pressure using a simplified form of the barometric formula. It takes into account the altitude of the observer.
- Check if the observer's altitude is greater than 11,000 meters. If so, it sets the temperature (temp) to 211.5 Kelvin. Otherwise, it calculates temp based on the given formula.
- Calculate the refraction angle R using the formula provided. This formula is used to correct for atmospheric refraction and is based on the U.S. Naval Observatory's Vector Astrometry Software. The formula takes the true altitude (alt_in_deg) and converts it to the apparent altitude after refraction.
- Handles a special case where alt_in_deg is exactly 90 degrees. In this case, a small correction (0.0019279) is added to R.
- Calculates a temperature and pressure correction factor (tpcor) based on the atmospheric pressure and temperature.
- Finally, it computes the corrected altitude by multiplying tpcor with R (converted from arcminutes to degrees) and adding it to the true altitude.

```python

def refraction(alt_in_deg, altitude):
    pres = 1010.*math.pow((1-6.5/288000*altitude),5.255)     #Simplified Barometric formula
    if (altitude>11000):
        temp = 211.5
    else:
        temp = 283.0 - (alpha * altitude)
    R = 1.02/math.tan((alt_in_deg + (10.3/(alt_in_deg + 5.11)))*(PI/180))  #This formula is used in the U. S. Naval Observatory's Vector Astrometry Software
    if (alt_in_deg == 90):
        R = R+ 0.0019279
    tpcor = (pres/1000) * (283/temp)
    R =  tpcor*(R/60)             #converting from arcminutes to degrees
    return alt_in_deg +R
```

- Obs --> It defines the observer's location, date, and time for astronomical observations.
- t --> It sets the observer's date and time.
- lst --> This line calculates the local apparent sidereal time (LST) at the given observer's location and time (t). (LST is the angle in the sky measured eastward from the vernal equinox and is used in astronomy to determine the right ascension of celestial objects.)

```python
Obs = ephem.Observer()
t = Time(Time_in_UTC, scale='utc',location= EarthLocation(lat = latitude*u.deg, lon=longitude*u.deg, height=altitude*u.m))
lst = t.sidereal_time('apparent')
print(lst)
```

1.3. Getting Recombination lines:

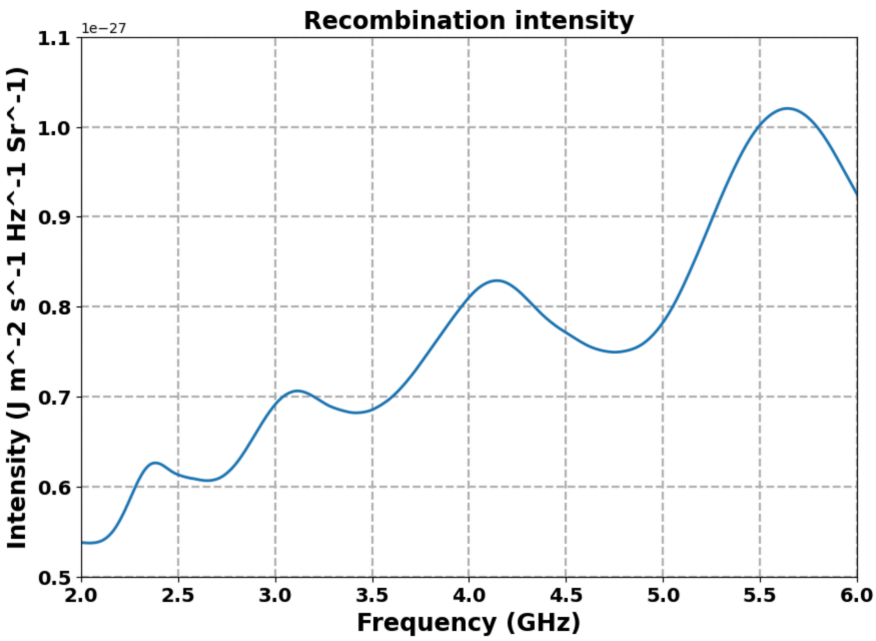
We plot the recombination lines. We then find the values of the recombination spectra for each of the channels of our interest.

- `recomb_file` is the full path to the data file 'total_spec_new.txt', located in the directory specified by recomb_path.
- `pd.read_csv(recomb_file, sep=" ", header=None)` reads the file into a pandas DataFrame. The sep=" "specifies that columns are separated by multiple spaces.
- `df.reset_index(drop=True, inplace=True)` resets the index of the DataFrame to default integer indices, dropping the original index.
- `df.columns = ["Frequency", "Intensity"]` sets the column names of the DataFrame to "Frequency" and "Intensity".
- "`x`" and "`y`" is a pandas Series containing the frequency values from the DataFrame.
- `InterpolatedUnivariateSpline` from the scipy.interpolate module is used to create an interpolation function based on the provided frequency (x) and intensity (y) data.
- `ius(frequencies)` applies the interpolation function to the array frequencies, generating interpolated intensity values at these new frequency points.
- `Rec_int` is an array of interpolated intensity values corresponding to the frequencies in the frequencies array.

```
recomb_file = os.path.join(recomb_path,'total_spec_new.txt')
df = pd.read_csv(recomb_file,sep= "    " ,header = None)
df.reset_index(drop=True, inplace=True)
df.columns = ["Frequency","Intensity"]

x = df["Frequency"]
y= df["Intensity"]

plt.figure(figsize=(10, 7))
plt.plot((df['Frequency']), (df['Intensity']),linewidth=2)
plt.title('Recombination intensity', fontsize='xx-large', fontweight='bold')
plt.xlabel('Frequency (GHz)', fontsize='xx-large', fontweight='bold')
plt.ylabel('Intensity (J m^-2 s^-1 Hz^-1 Sr^-1)', fontsize='xx-large', fontweight='bold')
plt.xlim([2,6])
plt.ylim([0.5e-27,1.1e-27])
plt.xticks(fontsize = 14, fontweight='bold')
plt.yticks(fontsize = 14, fontweight='bold')
plt.grid(linewidth=1.5, linestyle='--')
plt.show()

ius = InterpolatedUnivariateSpline(x, y)
Rec_int = ius(frequencies)
```

- `x_new` is a numpy array of 252 evenly spaced values between 2 and 6. These values are the new x-values where we want to estimate the corresponding y-values using interpolation.
- `interp1d` is a function from the scipy.interpolate module that creates an interpolation function. x and y are the original data points. These arrays contain the original x and y values that we want to interpolate between. `kind='linear'` specifies that we are using linear interpolation. This means the interpolated values will be on a straight line between the original data points.

```python
from scipy.interpolate import interp1d

x_new = np.linspace(2, 6, 252)

f = interp1d(x, y, kind='linear', fill_value='extrapolate')
y_new = f(x_new)

output_file = os.path.join(recomb_path, 'interpolated_values.txt')
with open(output_file, 'w') as f:
    for freq, inten in zip(x_new, y_new):
        f.write(f"{freq}\t{inten}\n")

file_path = '/Users/hshishir/Desktop/Theory/interpolated_values.txt'

recomb_li = pd.read_csv(file_path, delimiter='\t', header=None)

x = recomb_li[0]
y = recomb_li[1]

plt.figure(figsize=(10, 6))
plt.plot(x, y, marker='o', linestyle='-', color='b')
plt.xlabel('Frequency (GHz)')
plt.ylabel('Intensity (J m^-2 s^-1 Hz^-1 Sr^-1)')
plt.title('Interpolated Recombination Intensity')
plt.grid(True)
plt.show()

print(recomb_li.iloc[125,1])
```



`fill_value='extrapolate'` allows the interpolation function to estimate values outside the range of the original data points by extending the linear trend.

We convert the recombination spectra above to temperature units using Rayleigh Jean's approximation for each of the frequency channels, using the formula,

$$B_\nu = \frac{I_\nu c^2}{2k\nu^2} \ldots 2$$

```python
file_path = '/Users/hshishir/Desktop/Theory/interpolated_values.txt'
output_file_path = '/Users/hshishir/Desktop/Theory/brightness_temperature.txt'

recomb_li = pd.read_csv(file_path, delimiter='\t', header=None)

frequencies = recomb_li[0]
intensities = recomb_li[1]

brightness_temps = []

for j in range(252):
    brightness_temp_recomb = (intensities[j] * (c**2)) / (2 * k * 1e+18 * (frequencies[j] ** 2))
    brightness_temps.append((frequencies[j], brightness_temp_recomb))

with open(output_file_path, 'w') as file:
    for freq, temp in brightness_temps:
        file.write(f"{freq}\t{temp}\n")

brightness_temp_data = pd.read_csv(output_file_path, delimiter='\t', header=None)
frequencies_bt = brightness_temp_data[0]
brightness_temps_bt = brightness_temp_data[1]

plt.figure(figsize=(10, 6))
plt.plot(frequencies_bt, brightness_temps_bt, marker='o', linestyle='-', color='b')
plt.xlabel('Frequency (GHz)')
plt.ylabel('Brightness Temperature (K)')
plt.title('Brightness Temperature of Recombination Intensity')
plt.grid(True)
plt.show()

#print(brightness_temps_bt[125])
```

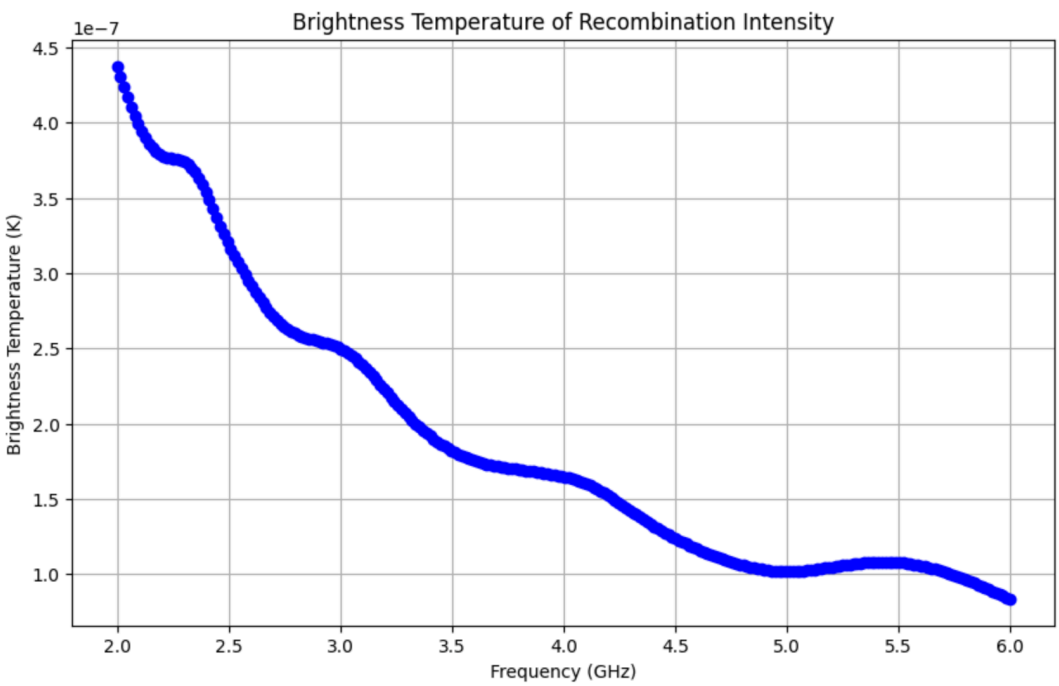


FIG 1: Brightness Temparature vs Frequency of Recombination lines

We describe a cell to calculate the CMB_int given by, $\frac{2h\nu^3}{c^2(e^{\frac{h\nu}{kT}}-1)}$ and P_hot_cold and P_dif.

```
CMB_Int = ((2*h*1e+27*frequencies**3)/(c*c))/(np.exp((h*frequencies*1e+9)/(k*TCMB))-1)
print("Got CMB Int!")

#P_hot and P_cold for calibration
P_hot = (2*h*frequencies*1e9)/(np.exp((h*frequencies*1e9)/(k*T_hot))-1)
P_cold = (2*h*frequencies*1e9)/(np.exp((h*frequencies*1e9)/(k*T_cold))-1)
P_dif = P_hot -P_cold

# Output the results for verification
print("CMB_Int:", CMB_Int)
print("P_hot:", P_hot)
print("P_cold:", P_cold)
print("P_dif:", P_dif)

print("Got P_dif for calibration!")
```

1.4. Getting Spectra:

The cell below generates the spectras. The general outline of the code is such that the first loop runs through each of the pixels and the second loop runs through each of the frequency channels for that particular pixel. The property of the sky is such that, say if we look at a particular pixel for three frequency channels and plot the Brightness temperature vs Frequency, then it will go by a power law, i.e., $T \propto \nu^{\alpha_{pixel}}$ (as shown is Figure ___). We will now look at this code is three parts.

Part 1:

- `hp.pixelfunc.pix2ang(16, i, nest=True, lonlat=True)` converts the pixel index i to angular coordinates (longitude l and latitude b) using HEALPix with NSIDE = 16 (NPIX = 12 x NSIDE^2) and nested indexing. `SkyCoord(l=coords[0]*u.degree, b=coords[1]*u.degree,frame='galactic')` creates a SkyCoord object representing the position in the Galactic coordinate system.
- `gc.transform_to(AltAz(obstime=Time_in_UTC[t], location=EarthLocation(lat=latitude*u.deg, lon=longitude*u.deg, height=altitude*u.m)))` transforms the Galactic coordinates to Altitude-Azimuth coordinates at a specific time (Time_in_UTC[t]) and observer location (latitude, longitude, altitude).
- alt = equi.alt.value extracts the altitude value from the transformed coordinates.
- new_alt = refraction(alt, altitude) corrects the altitude for atmospheric refraction effects.
- `func(xx, a)` defines a power law function where xx is the frequency and a is the spectral index. This function models the sky temperature at different frequencies based on the 408 MHz sky temperature (sky_408).
- `xx` is an array of the frequencies (in GHz) at which sky temperature data is available: 408 MHz (0.408 GHz), 1.42 GHz, and 23 GHz. `yy` is an array of sky temperature values at these frequencies for the current pixel i: sky_408[i-1], sky_1420[i-1], and sky_23000[i-1].
- `curve_fit(func, xx, yy)` fits the power law function func to the data points (xx, yy), determining the best-fit value of the spectral index a.

- `popt` contains the optimal values of the parameters (in this case, the spectral index a).
- `pcov` is the estimated covariance of `popt`, providing an estimate of the uncertainty in the parameter values.

Part 2:

We ultimately want to calculate the $T(\hat{r}_0, \nu, t) = \frac{\int_\Omega A(\hat{r}', \nu)\, T(\hat{r}', \nu)_{sky}\, d\hat{r}'}{\int_\Omega A(\hat{r}', \nu)\, d\hat{r}'}$, where "A" is the antenna gain and the denominator is the normalization factor. This part of the code helps us convert the existing maps into flux density, convolves with the beam pattern and obtains the power for each of the frequency channels and ultimately converts it into temperature again.

- `cwt = beam_def(freqs[j], new_alt, longitudes)` computes the beam weight for the current frequency freqs[j], altitude new_alt, and longitude longitudes.
- `map_1` computes the map intensity using the Planck constant k, speed of light c, frequency freqs[j], sky temperature at 408 MHz sky_408[i-1], and the spectral index `popt` (used to fit it in accordance to a power law).
- `Power` is calculated by summing the map intensity, CMB intensity CMB_Int[j], and recombination line intensity Rec_int[j], normalized by the frequency squared.
- `final_temp` computes the final temperature using the power, power differential P_dif[j], and the difference between hot and cold load temperatures T_hot and T_cold. (Look at the flow chart)
- `cross_real[j]` accumulates the weighted final temperature for the current channel j, accounting for the beam weight and a factor involving `gamma_func(freqs[j])`.
- `sumwt[j]` accumulates the beam weight `cwt` for normalization.

Part 3:

Here we used the *radiometer equation* to find the appropriate $\sigma = \frac{(T_{crossreal}(\hat{r}', t) + T_{rx} + T_{atm})}{\sqrt{\Delta\nu\Delta t}}$, where $\Delta\nu$ is the channel width and Δt is the integration time. Which helps us add noise to the system. We can note that an increase in the integration time helps us reduce the σ by a power of 1/2.

- `cross_real` is an array containing the accumulated real part of the spectra.
- A Gaussian noise array is generated with a mean of 0 and a standard deviation of 1 for each channel (No_of_Channels).
- The real part of the spectra (`cross_real`) is updated by adding the product of the noise array and the noise standard deviation sigma. Similarly, the imaginary part of the spectra (`cross_imag`) is updated in the same way.
- The noisy real part of the spectra is stored in `spectra` at index `t`. The noisy imaginary part of the spectra is stored in `spectra_cross_imag` at index `t`.

Then we ultimately plot each of the n_spectras (Brightness Temperature vs Frequency) for a particular LST.

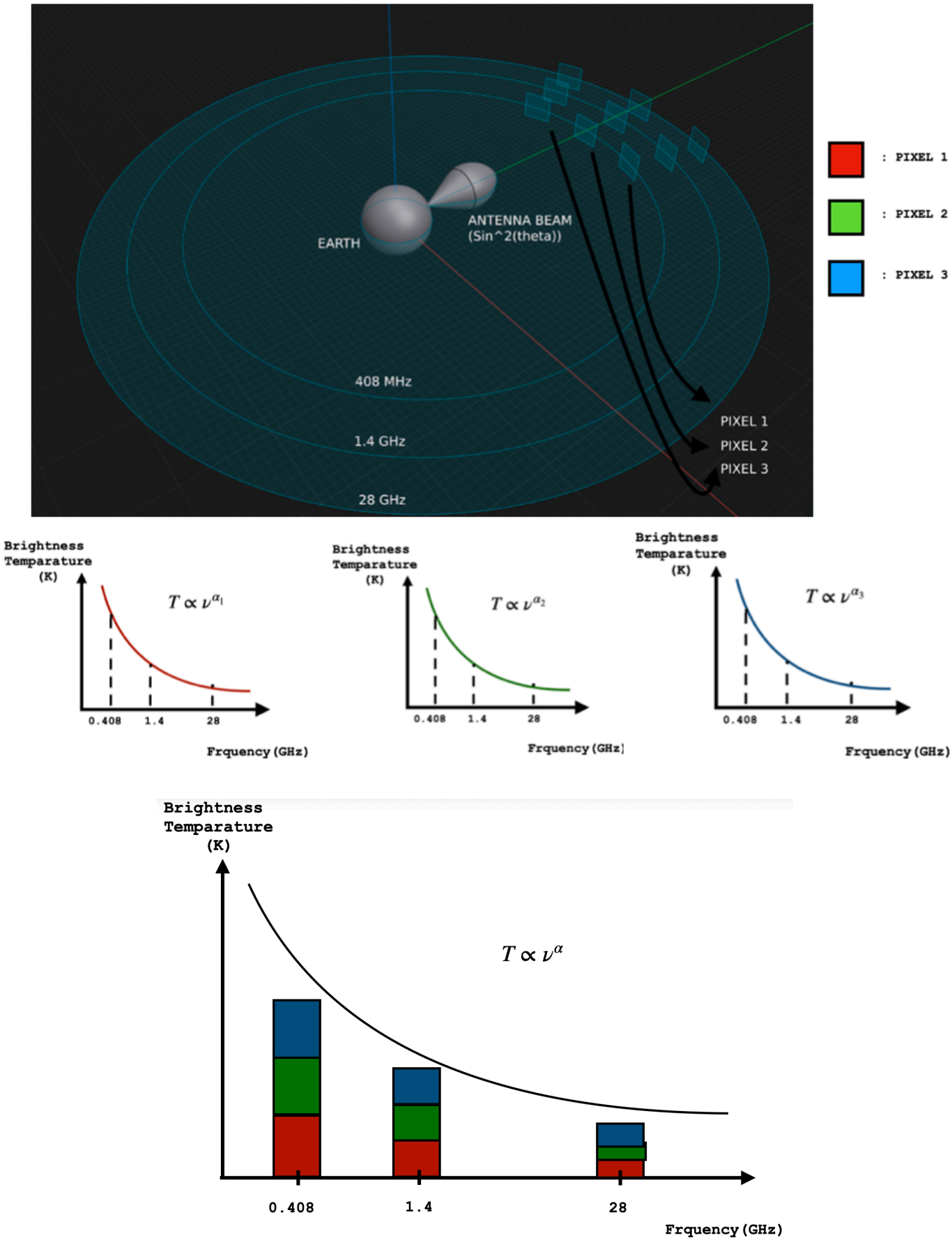


FIG 2: Pictorial representatiion of the code below

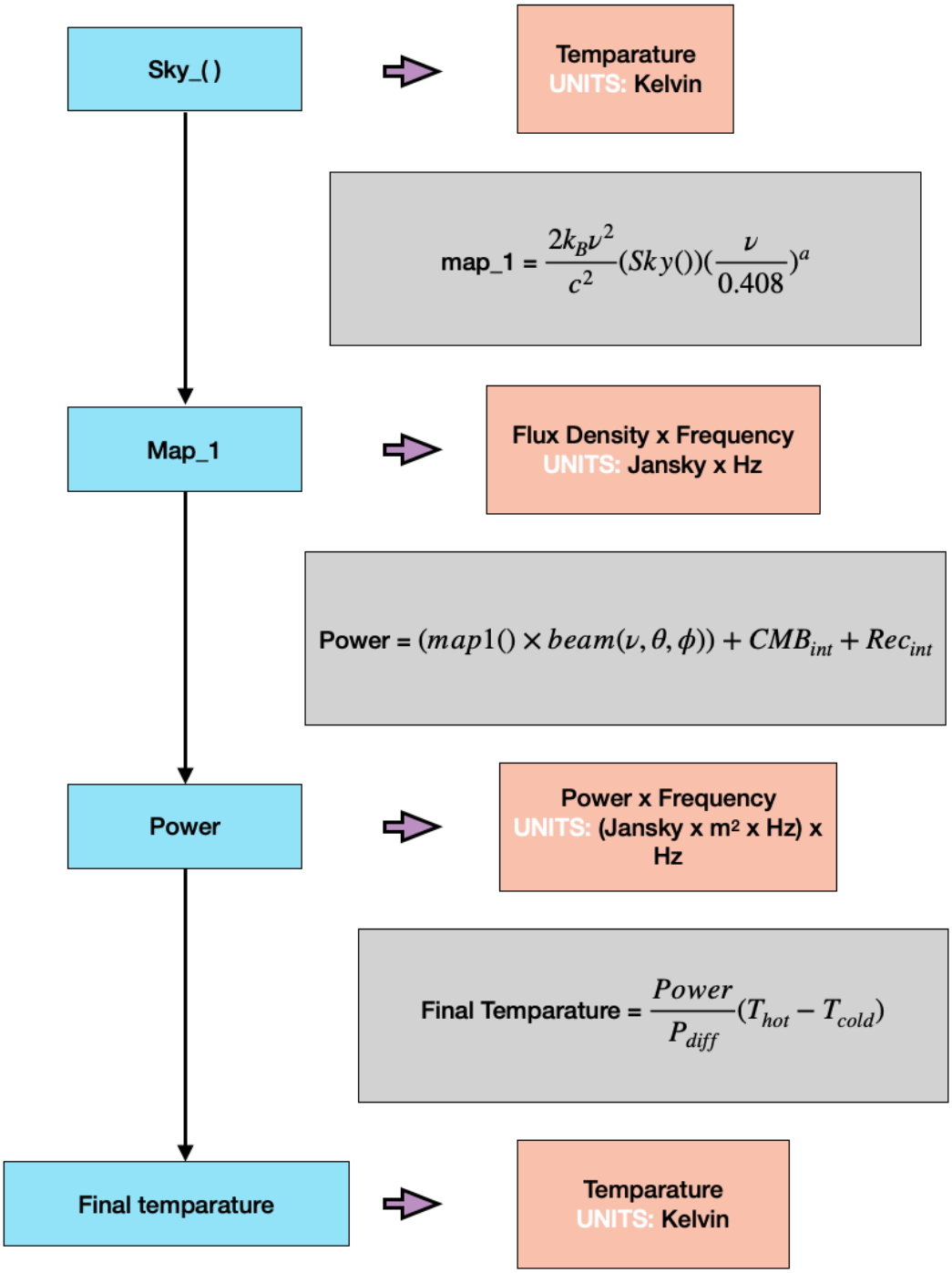


FIG 3: Flow chart for units conversion of the code below

```python
t = 0
while t<n_spectra:
    i=1
    while i<NPIX:            #Working on each pixel
        coords = hp.pixelfunc.pix2ang(16, i , nest = True, lonlat = True)          #Gives l and b coordinates for each pixel
        gc = SkyCoord(l = coords[0]*u.degree, b= coords[1]*u.degree, frame = 'galactic')

        #Now, the coordinates in Galactic frame is changed to equitorial frame below
        equi = gc.transform_to(AltAz(obstime = Time_in_UTC[t], location = EarthLocation(lat = latitude*u.deg, lon=longitude*u.deg , height=altitude*u.m)))
        alt = equi.alt.value
        new_alt = refraction(alt, altitude)

        #Using power law relation to fit spectra of three frequency maps
        def func(xx, a):
            return (sky_408)[i-1] * np.power((xx/0.408), a)

        xx = np.array([0.408, 1.420, 23])
        yy = np.array([sky_408[i-1], sky_1420[i-1], sky_23000[i-1]])

        popt, pcov = curve_fit(func, xx, yy)


        #print(freqs)
        j=0
        longitudes = equi.az.value
        while j< No_of_Channels:
            if (longitudes>359.00):
                longitudes = 359.00
            cwt = beam_def(freqs[j], new_alt, longitudes)
           # print(new_alt,cwt)
            if (cwt>0):     #Adding all the three intensities

                map_1 = (2*k*1e+18 * (freqs[j]**2) *sky_408[i-1]* np.power(freqs[j]/0.408, float(popt)))/ (c**2)
                Power = (map_1 + CMB_Int[j]+Rec_int[j]) *(c*c/ (1e+18* freqs[j]**2)) #+Rec_int[j]
                final_temp = (Power/P_dif[j])*(T_hot-T_cold)

                cross_real[j] = cross_real[j] + cwt*final_temp* (1 - np.power(gamma_func(freqs[j]),2))

                ## Remove the (1 - np.power(gamma_func(freqs[j]),2)) to try for ideal antenna (no return loss

                sumwt[j] = sumwt[j]+ cwt

            j = j+1

        if (i%1000 == 0):
            print('Number of pixels got: ', i)

        i = i+1


    i=0
    while(i< No_of_Channels):
        if sumwt[i]>0:
```

```
            cross_real[i] = cross_real[i]/sumwt[i]
        i = i+1


    sigma = (np.array(cross_real) + Trx+ T_atm)/ math.sqrt(Channel_width*1e9*Noise_int_time)

    #Generating an array of random integers with mean 0 and std 1.
    array = np.random.normal(0,1,No_of_Channels)

    #Adding the gaussian error

    cross_real = cross_real + array*sigma
    #Comment out this line if you need spectra without noise. To vary the noise level, change
    Noise_int_time in definitions

    cross_imag = cross_imag + array*sigma

    spectra[t] = cross_real
    spectra_cross_imag[t] = cross_imag

    cross_real = [0]*No_of_Channels
    cross_imag = [0]*No_of_Channels
    sumwt = [0]*No_of_Channels
    sigma = [0]*No_of_Channels

    print("No. of spectra got = ", t+1)

    t = t+1

Obs = ephem.Observer()
t = Time(Time_in_UTC, scale='utc',location=(longitude, latitude))
lst = t.sidereal_time('apparent')


i=0
while(i<n_spectra):
    plt.plot(freqs, spectra[i])
    i=i+1

plt.xlabel('Frequency [GHz]')
plt.ylabel('Temperature [K]')
plt.title('Simulated Sky Spectrum at LST :%s'%lst)
plt.grid()
plt.show()

plt.plot(freqs, spectra[0])
plt.xlabel('Frequency [GHz]')
plt.ylabel('Temperature [K]')
plt.title('One spectrum - Simulated Sky Spectrum at LST :%s'%lst[0])
plt.grid()
plt.show()
```

2. APSERa Simulation Pipeline for Cross Orthogonal Polarization Updated Code:

To access the polarization maps I, Q and U (as pysm doesn't generate V maps) we first import pysm. Just to clear up confusion pygdsm (pygsm) can only produce all sky intensity maps, i.e., stokes I maps.

The command `pysm3.Sky(nside=None, max_nside=None, preset_strings=["d2", "s1", "a1"], component_config=None, component_objects=None, output_unit=Unit("uK_RJ"), map_dist=None)`. By default, the `nside` is set to 512 whereas here we mostly use 64 or 32 (needs to be a power of 2). In the `present string`, "d2" takes into account polarization due to *dust*, "s1" is *synchrotron* and "a1" is *AME*. For our case, we only consider contributions due to synchrotron radiation.

```
import warnings
warnings.filterwarnings("ignore")

sky = pysm3.Sky(nside=64, preset_strings=["s1"])

sky.components
```

The next part of the code is used to produce Stokes I map using sky.get_emission() to produce the maps at a particular frequency. We store these maps for all the 252 channels in both fits and text format. We write a similar code for the Stokes Q and U maps.

```
gsm_2016 = GlobalSkyModel16()

fits_output_dir = "/Users/hshishir/Desktop/polarization_maps/I1/fits_file(64)"
txt_output_dir = "/Users/hshishir/Desktop/polarization_maps/I1/maps_tx(64)"

os.makedirs(fits_output_dir, exist_ok=True)
os.makedirs(txt_output_dir, exist_ok=True)

for freq in frequencies:
    print(f"Processing {freq} GHz...")

    map_2GHz = sky.get_emission(freq * u.GHz)
    map_2GHz_0 = map_2GHz[0]

    # Save the map as a FITS file
    fits_file_path = os.path.join(fits_output_dir, f"map_{freq}.fits")
    hp.write_map(fits_file_path, map_2GHz_0, overwrite=True)

    # Read the map from the FITS file (to ensure compatibility)
    sky_map_tx = hp.read_map(fits_file_path)

    # Save the map as a text file
    txt_file_path = os.path.join(txt_output_dir, f"map_{freq}.txt")
    np.savetxt(txt_file_path, sky_map_tx)

    print(f"Completed {freq} GHz.")

print("All maps have been processed and saved!")
```

We now access these files for each of the frequencies which stores the values of flux density for each of the pixels and form an array wherein each column corresponds to a particular frequency and a particular row corresponds to a certain pixel number.

```
maps = []
combined_output_dir = "/Users/hshishir/Desktop/polarization_maps/I2/maps_tx(32)"
txt_output_dir = "/Users/hshishir/Desktop/polarization_maps/I2/maps_tx(32)"

for freq in frequencies:
    txt_file_path = os.path.join(txt_output_dir, f"map_{freq}.txt")
    maps.append(np.loadtxt(txt_file_path))

combined_array = np.column_stack(maps)

combined_txt_path = os.path.join(combined_output_dir, 'combined_maps.txt')
np.savetxt(combined_txt_path, combined_array)

print("All maps have been processed and saved!")
print("Combined Array:\n", combined_array)
```

Iterate over each frequency in the `frequencies` list. Construct the file path for each frequency's map using `os.path.join`. Load the map from the text file using `np.loadtxt` and append it to the `maps list`. Combine the individual maps into a single array by stacking them column-wise using `np.column_stack`. This results in an array where each column corresponds to a different frequency's map. We write a similar code for the Stokes Q and U maps.

```
maps = []
combined_output_dir = "/Users/hshishir/Desktop/polarization_maps/I2/maps_tx(32)"
txt_output_dir = "/Users/hshishir/Desktop/polarization_maps/I2/maps_tx(32)"

for freq in frequencies:
    txt_file_path = os.path.join(txt_output_dir, f"map_{freq}.txt")
    maps.append(np.loadtxt(txt_file_path))

combined_array = np.column_stack(maps)

combined_txt_path = os.path.join(combined_output_dir, 'combined_maps.txt')
np.savetxt(combined_txt_path, combined_array)

print("All maps have been processed and saved!")
print("Combined Array:\n", combined_array)
```

2.1. Stokes Parameters:

We now look in brief and define the Stokes parameters. The relationship of the Stokes parameters I, Q, U and V to intensity and polarization ellipse which essentially forms the parameter space of Poincare sphere (which I found to be a very nice way to visualize Stokes parameters) are given by,

$I = I_0$

$$Q = I_0 p cos(2\psi) cos(2\chi)$$
$$U = I_0 p sin(2\psi) cos(2\chi)$$
$$V = I_0 p sin(2\chi) \ldots 1$$

where p is the degree of polarization, constrained by $0 \leq p \leq 1$. The factor of two before ψ represents the fact that any polarization ellipse is indistinguishable from one rotated by 180°, while the factor of two before χ indicates that an ellipse is indistinguishable from one with the semi-axis lengths swapped accompanied by a 90° rotation. It is important to note that the Stokes parameters have no memory of the initial phase.

The parametrized equations for Poincare sphere is given by,

$$I_0 = I$$
$$p = \frac{\sqrt{Q^2 + U^2 + V^2}}{I}$$
$$2\psi = tan^{-1}(\frac{U}{Q})$$
$$2\chi = tan^{-1}(\frac{V}{\sqrt{Q^2 + U^2}}) \ldots 2$$

The Stokes parameters can also be represented using the electric field components as,

$$I = E_x^2 + E_y^2$$
$$Q = E_x^2 - E_y^2$$
$$U = 2E_x E_y cos(\delta)$$
$$V = 2E_x E_y sin(\delta) \ldots 3$$

where E_x and E_y are the x and y components of electric fields respectively. δ is the phase difference between the two orthogonal components of the electric field. The polarization matrix can also be represented as $(I, Q, U, V)^T = (S_0, S_1, S_2, S_3)^T$.

Example:

TABLE 1: Stokes and Jones vectors for some polarization states:

State of Polarization	Stokes Vectors	Jones vectors
Horizontally polarized	$\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

State of Polarization	Stokes Vectors	Jones vectors
Vertically polarized	$\begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
+45⁰ polarized	$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
-45⁰ polarized	$\begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$
RH circularly polarized	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}$
LH circularly polarized	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix}$
Unpolarized	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	-

2.2. Simulation of Global Signal Observations:

For a particular time t and a certain direction in the sky $\hat{r}_0$, the sky averaged brightness temperature $T(\hat{r}_0, \nu, t)$ picked up by the antenna is given by,

$$T(\hat{r}_0, \nu, t) = \frac{\int_\Omega A(\hat{r}', \nu) T_{sky}(\hat{r}', \nu, t)\, d\hat{r}'}{\int_\Omega A(\hat{r}', \nu)\, d\hat{r}'} + T_N(\nu, t) \quad \ldots 1$$

where T_{sky} is the sky brightness temperature, A the antenna gain pattern, and T_N is the instrumental noise. We note that the antenna gain pattern is independent of time and that it is a property of the antenna and not the sky.

Now a cross-dipole antenna would be capable of measuring the polarization emissions for the sky. If we call s the intrinsic sky brightness distribution towards a line of sight $\hat{r}$ at the frequency ν in

terms of the usual Stokes parameters $\boldsymbol{s} = (I, Q, U, V)^{\mathrm{T}}$, the brightness observed by two orthogonal receptors $\boldsymbol{e} = (E_{xx}, E_{xy}, E_{yx}, E_{yy})^{\mathrm{T}}$ can be written as,

$$e(\hat{r}, \nu) = [J(\hat{r}, \nu) \otimes J^*(\hat{r}, \nu)]Ss(\hat{r}, \nu) \ldots 2$$

where J is the 2 × 2 Jones matrix representing the polarized receptor response and then we take the Kronneker product of the Jones matrix and its complex conjugate. A 'Stokes matrix' S is a coordinate transformation matrix in the 4 × 4 coherency domain: S^{+} transforms the representation from Stokes coordinates (I,Q,U,V) to linear polarisation coordinates (xx, xy, yx, yy). Similarly, $S^{\odot}$ transforms to circular polarisation coordinates (rr, rl, lr, ll). Following the convention of, we write:

$$S^{+} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & i \\ 0 & 0 & 1 & -i \\ 1 & -1 & 0 & 0 \end{bmatrix} \text{ and } S^{\odot} = \frac{1}{2}\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & i & 0 \\ 0 & 1 & -i & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix} \ldots 3$$

Let us consider the most general case of the Jones matrix with the leakage terms too,

$$J = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{bmatrix} \ldots 4$$

We refer to $[J(\hat{r}, \nu) \otimes J^*(\hat{r}, \nu)]S$ as the "mixing matrix". The general structure of the mixing matrix for the most generalized case of Jones matrix is given by,

$$[J(\hat{r}, \nu) \otimes J^*(\hat{r}, \nu)]S = \frac{1}{2}\begin{bmatrix} J_{xx}J^*_{xx} + J_{xy}J^*_{xy} & J_{xx}J^*_{xx} - J_{xy}J^*_{xy} & J_{xx}J^*_{xy} + J_{xy}J^*_{xx} & i(J_{xx}J^*_{xy} - J_{xy}J^*_{xx}) \\ J_{xx}J^*_{yx} + J_{xy}J^*_{yy} & J_{xx}J^*_{yx} - J_{xy}J^*_{yy} & J_{xx}J^*_{yy} + J_{xy}J^*_{yx} & i(J_{xx}J^*_{yy} - J_{xy}J^*_{yx}) \\ J_{yx}J^*_{xx} + J_{yy}J^*_{xy} & J_{yx}J^*_{xx} - J_{yy}J^*_{xy} & J_{yx}J^*_{xy} + J_{yy}J^*_{xx} & i(J_{yx}J^*_{xy} - J_{yy}J^*_{xx}) \\ J_{yx}J^*_{yx} + J_{yy}J^*_{yy} & J_{yx}J^*_{yx} - J_{yy}J^*_{yy} & J_{yx}J^*_{yy} + J_{yy}J^*_{yx} & i(J_{yx}J^*_{yy} - J_{yy}J^*_{yx}) \end{bmatrix}$$

$\ldots 5$

The mixing matrix is formed using the given code.

```python
def mixing_matrix(frequencies, alt, az):
    sin_theta = (np.sin(alt*np.pi/180))**2

    matrix = np.array([[sin_theta * 0.5, 0.0],
                       [0.0, sin_theta * 0.5]], dtype=complex)

    conjugate_matrix = np.conjugate(matrix)

    kronecker_product = np.kron(matrix, conjugate_matrix)

    S_matrix = np.array([[0.5, 0.5, 0, 0],
                         [0, 0, 0.5, 0.5*1j],
                         [0, 0, 0.5, -0.5*1j],
                         [0.5, -0.5, 0, 0]], dtype=complex)

    result = np.dot(kronecker_product, S_matrix)

    return result
```

Now using equation (2) we find that the general expression for E_{xx} as a linear combination of the stokes parameters to be,

$$E_{xx} = \frac{1}{2}[(J_{xx}J^*_{xx} + J_{xy}J^*_{xy})I + (J_{xx}J^*_{xx} - J_{xy}J^*_{xy})Q + (J_{xx}J^*_{xy} + J_{xy}J^*_{xx})U + i(J_{xx}J^*_{xy} - J_{xy}J^*_{xx})V] \quad \ldots 6$$

Equation (1) can be rewritten for all the polarization components as,

$$T_{xx}(\hat{r}_0, \nu, t) = \frac{\int_\Omega E_{xx}(\hat{r}', \nu)\, d\hat{r}'}{\int_\Omega A_{xx}(\hat{r}', \nu)\, d\hat{r}'} \quad \ldots 7$$

where A_{xx} is given by,

$$A_{xx} = \frac{1}{2}\sqrt{(J_{xx}J^*_{xx} + J_{xy}J^*_{xy})^2 + (J_{xx}J^*_{xx} - J_{xy}J^*_{xy})^2 + (J_{xx}J^*_{xy} + J_{xy}J^*_{xx})^2 - (J_{xx}J^*_{xy} - J_{xy}J^*_{xx})^2} \quad \ldots 8$$

Similarly,

$$T_{xy}(\hat{r}_0, \nu, t) = \frac{\int_\Omega E_{xy}(\hat{r}', \nu)\, d\hat{r}'}{\int_\Omega A_{xy}(\hat{r}', \nu)\, d\hat{r}'} \quad \ldots 9$$

where,

$$E_{xy} = \frac{1}{2}[(J_{xx}J^*_{yx} + J_{xy}J^*_{yy})I + (J_{xx}J^*_{yx} - J_{xy}J^*_{yy})Q + (J_{xx}J^*_{yy} + J_{xy}J^*_{yx})U + i(J_{xx}J^*_{yy} - J_{xy}J^*_{yx})V] \quad \ldots 10$$ and,

$$A_{xy} = \frac{1}{2}\sqrt{(J_{xx}J^*_{yx} + J_{xy}J^*_{yy})^2 + (J_{xx}J^*_{yx} - J_{xy}J^*_{yy})^2 + (J_{xx}J^*_{yy} + J_{xy}J^*_{yx})^2 - (J_{xx}J^*_{yy} - J_{xy}J^*_{yx})^2} \quad \ldots 11$$

Also,

$$T_{yx}(\hat{r}_0, \nu, t) = \frac{\int_\Omega E_{yx}(\hat{r}', \nu)\, d\hat{r}'}{\int_\Omega A_{yx}(\hat{r}', \nu)\, d\hat{r}'} \quad \ldots 12$$

where,

$$E_{yx} = \frac{1}{2}[(J_{yx}J^*_{xx} + J_{yy}J^*_{xy})I + (J_{yx}J^*_{xx} - J_{yy}J^*_{xy})Q + (J_{yx}J^*_{xy} + J_{yy}J^*_{xx})U + i(J_{yx}J^*_{xy} - J_{yy}J^*_{xx})V] \quad \ldots 13$$ and,

$$A_{yx} = \frac{1}{2}\sqrt{(J_{yx}J^*_{xx} + J_{yy}J^*_{xy})^2 + (J_{yx}J^*_{xx} - J_{yy}J^*_{xy})^2 + (J_{yx}J^*_{xy} + J_{yy}J^*_{xx})^2 - (J_{yx}J^*_{xy} - J_{yy}J^*_{xx})^2} \quad \ldots 14$$

Finally,

$$T_{yy}(\hat{r}_0, \nu, t) = \frac{\int_\Omega E_{yy}(\hat{r}', \nu)\, d\hat{r}'}{\int_\Omega A_{yy}(\hat{r}', \nu)\, d\hat{r}'} \ldots 15$$

where,

$$E_{yy} = \frac{1}{2}[(J_{yx}J^*_{yx} + J_{yy}J^*_{yy})I + (J_{yx}J^*_{yx} - J_{yy}J^*_{yy})Q + (J_{yx}J^*_{yy} + J_{yy}J^*_{yx})U + i(J_{yx}J^*_{yy} - J_{yy}J^*_{yx})V] \ldots 16$$

$$A_{yy} = \frac{1}{2}\sqrt{(J_{yx}J^*_{yx} + J_{yy}J^*_{yy})^2 + (J_{yx}J^*_{yx} - J_{yy}J^*_{yy})^2 + (J_{yx}J^*_{yy} + J_{yy}J^*_{yx})^2 - (J_{yx}J^*_{yy} - J_{yy}J^*_{yx})^2} \ldots 17$$

Example:

Now we can take an ideal Jones matrix with no leakage terms and set $V(\hat{r}, \nu, t) = 0\ \forall\ \hat{r}, \nu$. That is

$$J = sin^2(\theta)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \ldots 18$$

The matrix $e = (E_{xx}, E_{xy}, E_{yx}, E_{yy})^T$ is updated to

$$e(\hat{r}, \nu) = \frac{1}{2}\begin{bmatrix} I + Q \\ U \\ U \\ I - Q \end{bmatrix} \ldots 19$$

If we now define a matrix $A = (A_{xx}, A_{xy}, A_{yx}, A_{yy})^T$, which we call the "antenna gain" matix is now updated to,

$$A(\hat{r}, \nu) = \frac{1}{2}sin^2(\theta)\begin{bmatrix} 2 \\ 1 \\ 1 \\ 2 \end{bmatrix} \ldots 20$$

The above is implemented in the code as given in the circled segment. This updated code takes care of both real and imaginary part. The pysm maps are in the units of flux density, hence the convertion as given in the flowchart below.

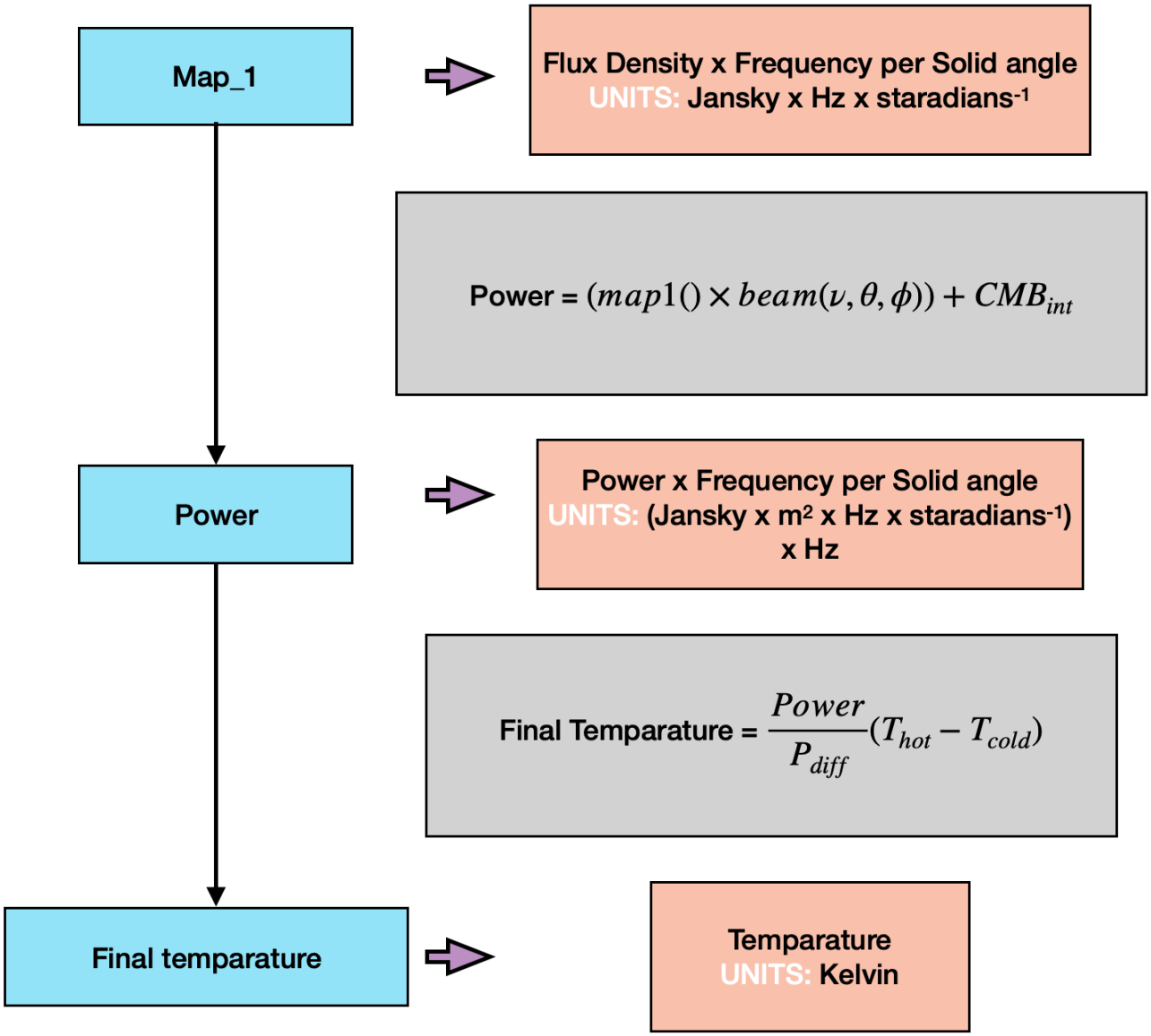


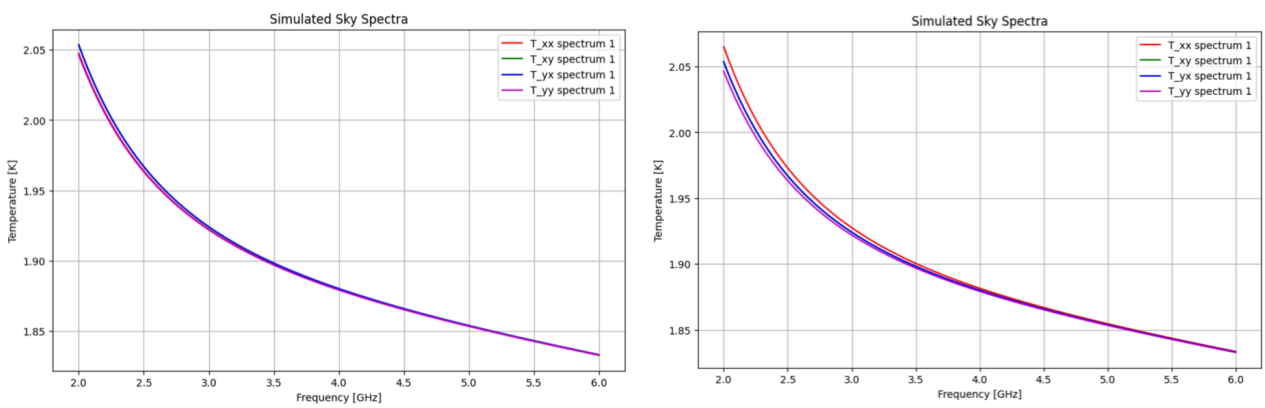
FIG 4: Flow chart for units conversion of the code below

```python
output_dir = '/Users/hshishir/Desktop/polarization_maps/halfwave1'
os.makedirs(output_dir, exist_ok=True)

titles = ["T_xx", "T_xy", "T_yx", "T_yy"]

for h in range(4):

    t = 0
    while t < n_spectra:
        i = 0
        while i < 12288:  # Working on each pixel
            coords = hp.pixelfunc.pix2ang(32, i, nest = True, lonlat=True)  # Gives l and b coordinates for each pixel
            gc = SkyCoord(l=coords[0] * u.degree, b=coords[1] * u.degree, frame='galactic')

            # Now, the coordinates in Galactic frame are changed to equatorial frame below
            equi = gc.transform_to(AltAz(obstime=Time_in_UTC[t], location=EarthLocation(lat=latitude * u.deg, lon=longitude * u.deg, height=altitude * u.m)))
            alt = equi.alt.value
            new_alt = refraction(alt, altitude)



            # Using power law relation to fit spectra of three frequency maps
            #def func(xx, a):
                #return sky_408[i - 1] * np.power((xx / 0.408), a)

            #xx = np.array([0.408, 1.420, 23])
            #yy = np.array([sky_408[i - 1], sky_1420[i - 1], sky_23000[i - 1]])

            #popt, pcov = curve_fit(func, xx, yy)
            #a_param = popt[0]

            j = 0
            longitudes = equi.az.value
            while j < 252:
                new_map = combined_array[:,j]
                new_map1 = combined_array1[:,j]
                new_map2 = combined_array2[:,j]
                if longitudes > 359.00:
                    longitudes = 359.00
                result = mixing_matrix1(frequencies[j], new_alt, None)
                cwt = np.sqrt((result[h,0]**2) + (result[h,1]**2) + (result[h,2]**2) + (result[h,3]**2))

                if cwt > 0:  # Adding all the three intensities
                    map_0 = (2 * k * 1e+18 * (frequencies[j] ** 2) * 1e-6 *new_map[i - 1]) / (c ** 2)
                    map_1 = (2 * k * 1e+18 * (frequencies[j] ** 2) * 1e-6 *new_map1[i - 1]) / (c ** 2)
                    map_2 = (2 * k * 1e+18 * (frequencies[j] ** 2) * 1e-6 *new_map2[i - 1]) / (c ** 2)
                    Power = (( ((map_0 + CMB_Int[j] + recomb_li.iloc[j,1]) * result[h,0]) + (map_1 * result[h,1]) + (map_2 * result[h,2]) + (0 * result[h,3]) ) ) * (c * c / (1e+18 * frequencies[j] ** 2))  # +Rec_int[j]
                    final_temp = (Power / P_dif[j]) * (T_hot - T_cold)
                    cross_mix[j] = cross_mix[j] + 1 * final_temp
                    sumwt[j] = sumwt[j] + cwt
```

```
                j = j + 1

            if i % 10000 == 0:
                print('Number of pixels got: ', i)

            i = i + 1

        i = 0
        while i < No_of_Channels:
            if sumwt[i] > 0:
                cross_mix[i] = cross_mix[i] / sumwt[i]  # normalization with respect to the beam pattern
            i = i + 1

        cross_mix = np.array(cross_mix)
        cross_real = cross_mix.real
        cross_imag = cross_mix.imag

        sigma_real = (np.array(cross_real) + Trx + T_atm) / np.sqrt(Channel_width * 1e+9 * Noise_int_time)  # radiometric equation
        sigma_imag = (np.array(cross_imag) + Trx + T_atm) / np.sqrt(Channel_width * 1e+9 * Noise_int_time)

        # Generating an array of random integers with mean 0 and std 1.
        array = np.random.normal(0, 1, No_of_Channels)

        # Adding the gaussian error
        cross_real = cross_real + array * sigma_real
        cross_imag = cross_imag + array * sigma_imag

        output_file_real = os.path.join(output_dir, f"spectrum_CRR_real{titles[h]}_{t}.txt")
        with open(output_file_real, "w") as file:
            for freq, temp in zip(frequencies, cross_real):
                file.write(f"{freq}\t{temp}\n")

        output_file_imag = os.path.join(output_dir, f"spectrum_CRR_imag{titles[h]}_{t}.txt")
        with open(output_file_imag, "w") as file:
            for freq, temp in zip(frequencies, cross_imag):
                file.write(f"{freq}\t{temp}\n")


        spectra_real[t] = cross_real
        spectra_imag[t] = cross_imag

        cross_real = [0] * No_of_Channels
        cross_imag = [0] * No_of_Channels
        sumwt = [0] * No_of_Channels
        sigma = [0] * No_of_Channels

        print("No. of spectra got = ", t + 1)

        t = t + 1

    Obs = ephem.Observer()
    t = Time(Time_in_UTC, scale='utc', location=(longitude, latitude))
    lst = t.sidereal_time('apparent')

    i = 0
    while i < n_spectra:
        plt.plot(frequencies, spectra_real[i])
        i = i + 1
```

```python

    plt.xlabel('Frequency [GHz]')
    plt.ylabel('Temperature [K]')
    plt.title(f"Simulated Sky Spectrum at LST for {titles[h]} real :%s" % lst)
    plt.grid()
    plt.show()

    print(f"h = {h} is completed")

    #plt.plot(frequencies, spectra_real[0])
    #plt.xlabel('Frequency [GHz]')
    #plt.ylabel('Temperature [K]')
    #plt.title('One spectrum - Simulated Sky Spectrum at LST for T_yy real :%s' % lst[0])
    #plt.grid()
    #plt.show()

    i = 0
    while i < n_spectra:
        plt.plot(frequencies, spectra_imag[i])
        i = i + 1

    plt.xlabel('Frequency [GHz]')
    plt.ylabel('Temperature [K]')
    plt.title(f"Simulated Sky Spectrum at LST for {titles[h]} imaginary :%s" % lst)
    plt.grid()
    plt.show()

    print(f"h = {h} is completed")

    #plt.plot(frequencies, spectra_imag[0])
    #plt.xlabel('Frequency [GHz]')
    #plt.ylabel('Temperature [K]')
    #plt.title('One spectrum - Simulated Sky Spectrum at LST for T_yy imaginary :%s' % lst[0])
    #plt.grid()
    #plt.show()
```

We now plot the graph for T_{xx}, T_{xy}, T_{yx} and T_{yy}. We consider two cases, one which doesn't includes the recombination lines and the other which includes the recombination lines.



A B

FIG 5: A) Spectra without Recombination int **B)** Spectra with Recombination int

We now take the difference between both the spectras to observe if we can obtain the recombination lines back (this can only be done for simulations, in actual observations we will have to use MS (Maximally smooth) fit functions).

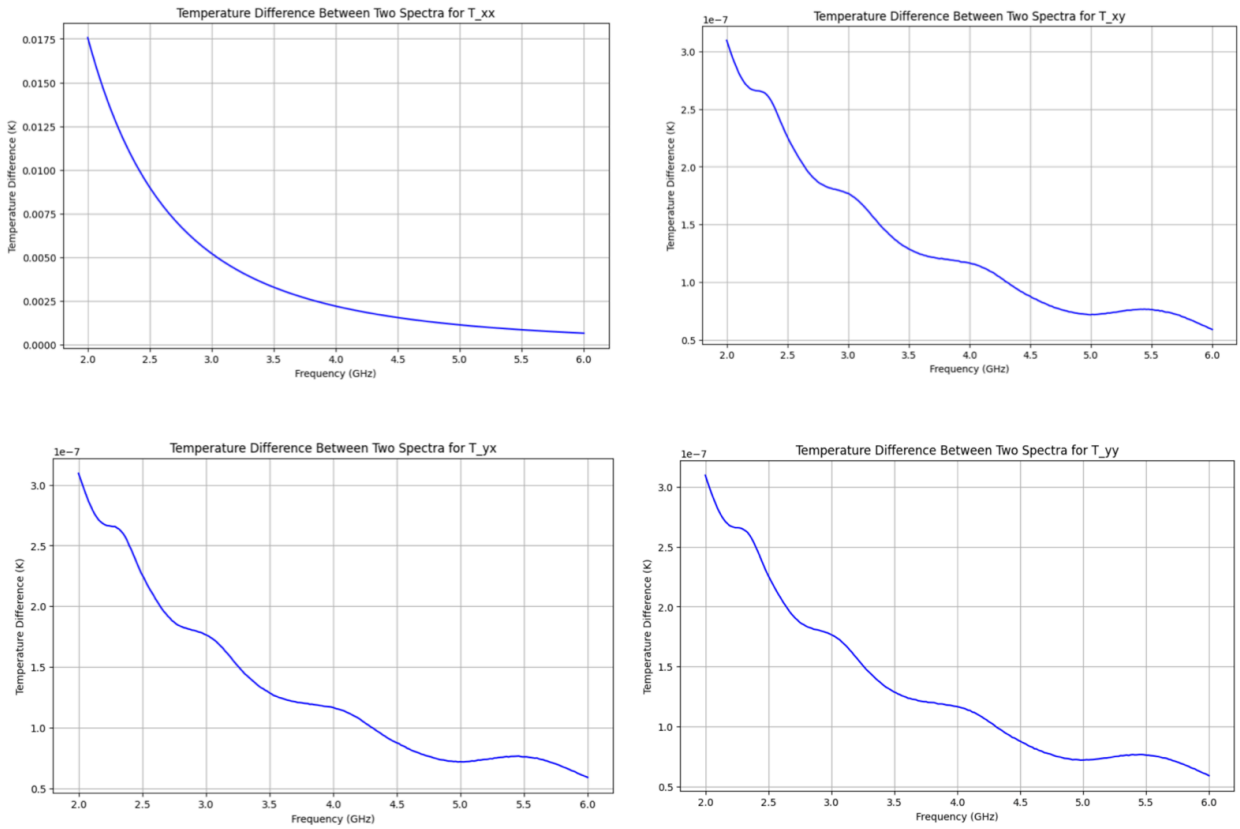


FIG 6: Difference between the two Spectra

2.3. Double Refraction and Reatarder Plates:

In the case of an unpolarized light, we can split it into two orthogonal components, E_x and E_y. The different polarization states of lights can be,

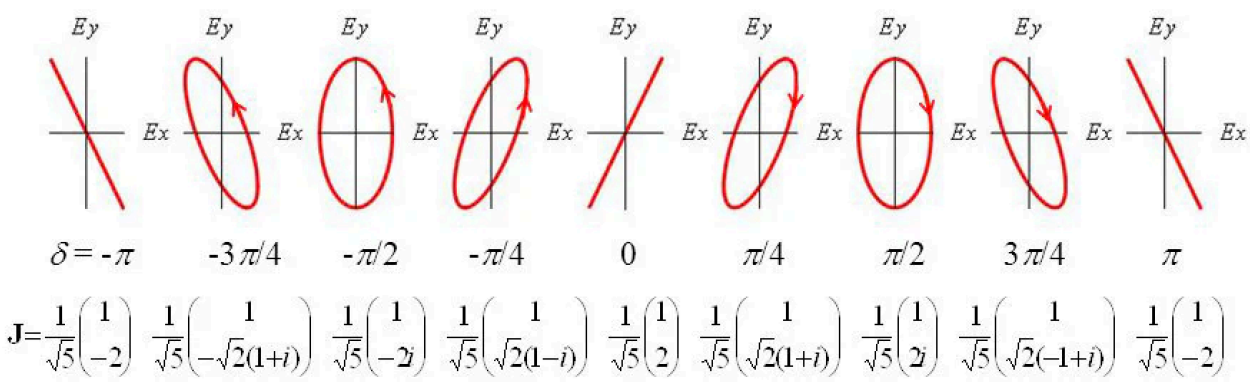


FIG 7: Example of Jones vectors

where δ is the phase difference between the E_x and E_y components of the electric field.

Let us now understand how different polarization states of an incoming electric field can be achieved using the phenomenon of double refraction. In our case, we consider a calcite crystal (which is a uniaxial, negative crystal).

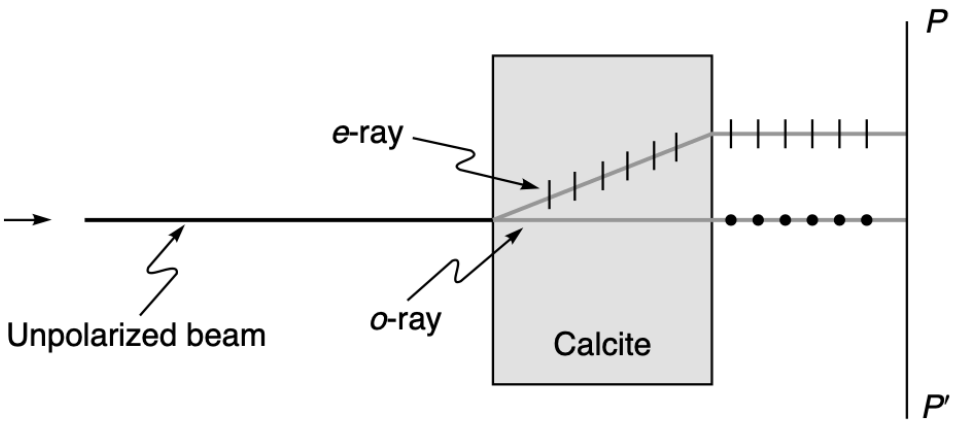


FIG 8: When an unpolarized light beam is incident normally on a calcite crystal, it would in general, split up into two linearly polarized beams

When an unpolarized light beam is incident normally on a calcite crystal, it would in general, split up into two linearly polarized beams. The beam that travels undeviated is known as the ordinary ray (o-ray) and obeys Snell's laws of refraction. On the other hand, the second beam, which in general does not obey Snell's laws, is known as the extraordinary ray (e-ray). The velocity of the ordinary ray is the same in all directions, the velocity of the extraordinary ray is different in different directions. Along a particular direction, the two velocities are equal; this direction is known as the optic axis of the crystal. In a crystal such as calcite, the two rays have the same speed only along one direction; such crystals are known as uniaxial crystals. The velocities of the ordinary and the extraordinary rays are given by the following equations,

$$v_{no} = \frac{c}{n_o} \text{ ordinary ray,}$$

$$\frac{1}{v_{re}^2} = \frac{sin^2(\theta)}{(\frac{c}{n_e})^2} + \frac{cos^2(\theta)}{(\frac{c}{n_o})^2} \text{ extraordinary ray } \ldots 1$$

where no and ne are constants of the crystal and θ is the angle that the ray makes with the optic axis.

1. Both ordinary and extraordinary waves are linearly polarized.

2. $\vec{D}\cdot\vec{k} = 0$ for both o- and e-waves. Thus D is always at right angles to k, and for this reason the direction of D is chosen as the direction of "vibrations."

3. If we assume the z-axis to be parallel to the optic axis then $\vec{D}\cdot\vec{z} = 0$ (and $\vec{D}\cdot\vec{k} = 0$) for the o-wave. Thus for the o-wave, the $\vec{D}$ is at right angles to the optic axis as well as to k.

4. On the other hand, for the e-wave, D lies in the plane containing k and the optic axis, and of course, $\vec{D}\cdot\vec{k} = 0$.

Consider the image below to analyse the construct of the Quarter wave and Half wave plate. For a random state of polarization of the incident beam, both the extraordinary and the ordinary components will be present. For a negative crystal such as calcite $n_e < n_o$, the e-wave will travel faster than the o-wave. Such a beam can be assumed to be a superposition of two linearly polarized beams polarized along the y and z directions with amplitudes $E_0 sin(\phi)$ and $E_0 cos(\phi)$, respectively. The z component passes through as an extraordinary beam propagating with wave velocity $\frac{c}{n_e}$. The y component passes through as an ordinary beam propagating with wave velocity $\frac{c}{n_o}$. Since $n_e \neq n_o$ the two beams will propagate with different velocities; as such, when they come out of the crystal, they will not be in phase. Consequently, the emergent beam will be, in general, elliptically polarized.

Let the plane x = 0 represent the surface of the crystal on which the beam is incident. The y and z components of the incident beam can be written in the form,

$$E_y = E_0\ sin(\phi)\ cos(kx - \omega t)$$

$$E_z = E_0\ cos(\phi)\ cos(kx - \omega t) \ldots 2$$

where k ($= \frac{\omega}{c}$) represents the free space wave number. Thus, at x = 0, we have,

$$E_y(x = 0) = E_0\ sin(\phi)\ cos(\omega t)$$

$$E_z(x = 0) = E_0\ cos(\phi)\ cos(\omega t) \ldots 3$$

Inside the crystal, the two components will be given by,

$$E_y = E_0\ sin(\phi)\ cos(n_o kx - \omega t) \quad ordinary\ wave$$

$$E_z = E_0\ cos(\phi)\ cos(n_e kx - \omega t) \quad extraordinary\ wave \ldots 4$$

If the crystal has a thickness of "d" then the emergent wave takes the form,

$E_y = E_0\ sin(\phi)\ cos(\omega t - \theta_o)$

$E_z = E_0\ cos(\phi)\ cos(\omega t - \theta_e)$ ...5

where $\theta_o = n_o kd$ and $\theta_e = n_e kd$. By appropriately choosing the instant $t = 0$, the components may be rewritten as,

$E_y = E_0\ sin(\phi)\ cos(\omega t - \theta)$

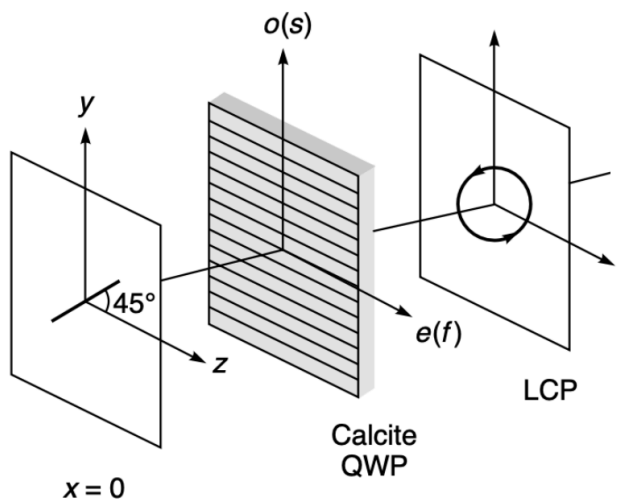
$E_z = E_0\ cos(\phi)\ cos(\omega t)$ ...6

where

$$\theta = \theta_o - \theta_e = kd(n_o - n_e) = \frac{\omega}{c}(n_o - n_e)d \ ...7$$

represents the phase difference between the ordinary and the extraordinary beams. If the thickness of the crystal is such that $\theta = 2\pi, 4\pi, \ldots$, the emergent beam will have the same state of polarization as the incident beam. Now, if the thickness d of the crystal is such that $\theta = \pi/2$, the crystal is said to be a ***Quarter wave plate***—a phase difference $\pi/2$implies a path difference of a quarter of a wavelength. On the other hand, if the thickness of the crystal is such that $\theta = \pi$, the crystal is said to be a ***Half wave plate***.

Example 1:



Let us consider the case when $\phi = \pi/4$ and $\theta = \pi/2$; i.e., the y and z components of the incident wave have equal amplitudes, and the crystal introduces a phase difference of $\pi/2$ as given below. Thus, for the emergent beam we have,

$$E_y = \frac{E_0}{\sqrt{2}} sin(\omega t) \qquad E_z = \frac{E_0}{\sqrt{2}} cos(\omega t) \ldots 8$$

This represents circular polarization as,

$$E_y^2 + E_z^2 = \frac{E_0^2}{\sqrt{2}} \ldots 9$$

To determine the direction of rotation of the electric vector, we note that at t = 0,

$$E_y = 0 \qquad E_z = \frac{E_0}{\sqrt{2}} \ldots 10$$

and at $t = \Delta t$,

$$E_y \approx \frac{E_0}{\sqrt{2}} \omega \Delta t \qquad E_z \approx \frac{E_0}{\sqrt{2}} \ldots 11$$

The above equations show that as time increases, the electric vector rotates in the counterclockwise direction, and hence the beam is left circularly polarized. To introduce a phase difference of $\pi/2$, the thickness of the crystal should have a value given by,

$$d = \frac{c}{\omega(n_e - n_e)} \frac{\pi}{2} = \frac{1}{4} \frac{\lambda_0}{n_o - n_e} \ldots 12$$

where λ_0 is the free space wavelength. For calcite,

$$n_o = 1.65836 \qquad n_e = 1.48641$$

which correspond to λ_0 = 5893 Å at 18°C. Substituting these values, we obtain

$$d = \frac{5893 \times 10^{-8}}{4 \times 0.17195} \; cm \approx 0.000857 \; mm$$

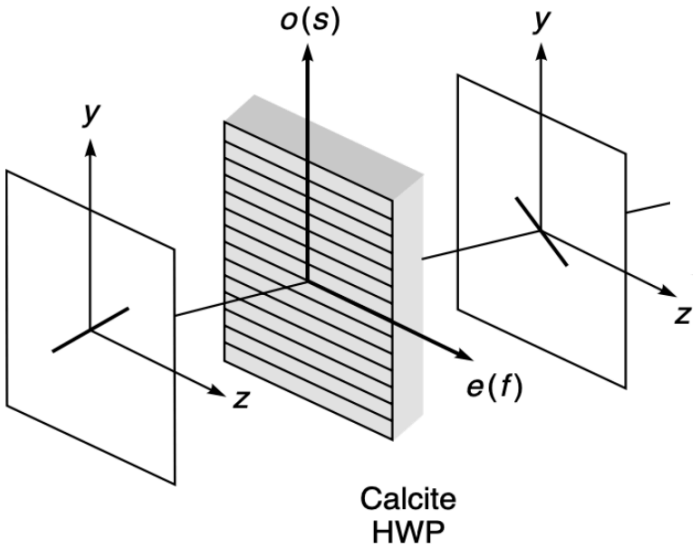
If the thickness is an odd multiple of the above quantity, i.e., if

$$d = (2m + 1) \frac{\lambda_0}{4(n_o - n_e)} \qquad m = 0,1,2,\ldots \ldots 13$$

then in the example considered above it can be easily shown that the emergent beam will be left circularly polarized for m = 0, 2, 4, . . . and right circularly polarized for m = 1, 3, 4, ...

Example 2:

We next consider the case when the linearly polarized beam (with $\phi = \pi/4$) is incident on a HWP so that $\theta = \pi$; i.e., the y and z components of the incident wave have equal amplitudes, and the crystal introduces a phase difference of π. Thus, for the emergent beam we have



$$E_y = -\frac{E_0}{\sqrt{2}} cos(\omega t) \qquad E_z = \frac{E_0}{\sqrt{2}} cos(\omega t) \ldots 14$$

which represents a linearly polarized wave with the direction of polarization making an angle of 135° with the z axis. If we now pass this beam through a calcite QWP, the emergent beam will be right circularly polarized. Thus, for a HWP the thickness (for a negative crystal) is given by,

$$d = (2m+1)\frac{\lambda_0}{2(n_o - n_e)} \ldots 15$$

If the crystal thickness is such that if $\theta \neq \pi/2, \pi, 3\pi/2, 2\pi, \ldots$, the emergent beam will be elliptically polarized.

Now moving away from the microscopic description of retarders, we look at Jones matrices of the same. Since we can write a polarization state as a vector, we use matrices, J, to transform them from the input polarization, E_0, to the output polarization, E_1. Jones matrix can be thought of as a transfer function,

$$E_1 = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{bmatrix} E_0 = JE_0 \ldots 16$$

Using the above we obtain,

$$E_{1x} = J_{xx}E_{0x} + J_{xy}E_{0y}$$

$$E_{1y} = J_{yx}E_{0x} + J_{yy}E_{0y} \ldots 17$$

TABLE 1: Jones Matrices of Standard components:

Polarizer Type	Jones Matirx
Horizontal Linear Polarizer	$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$
Vertical Linear Polarizer	$\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$
Linear Polarizer at +45^0	$\frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$
Linear Polarizer at -45^0	$\frac{1}{2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$
Quarter wave plate, fast axis verticle	$e^{i\pi/4}\begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$
Quarter wave plate, fast axis horizontal	$e^{i\pi/4}\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
Homogeneous circular polarizer right	$\frac{1}{2}\begin{bmatrix} 1 & i \\ -i & 1 \end{bmatrix}$
Homogeneous circular polarizer left	$\frac{1}{2}\begin{bmatrix} 1 & -i \\ i & 1 \end{bmatrix}$

When there are multiple retarder plates such as M_1, M_2. and M_3 then the transfer function for the case below is updated to,

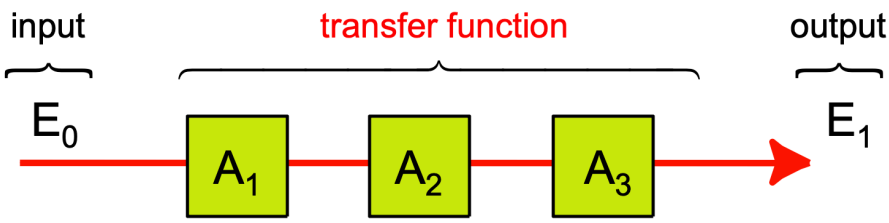


FIG 9: Using a series of Retarder plates

$S_{out} = M_3 M_2 M_1 S_{in} \ldots 18$

Now we introduce a quarter wave plate, i.e., the new jones matrix is updated to, $J' = [Quarter\ wave\ plate]J$.

```python
def mixing_matrix1(frequencies, alt, az):
    sin_theta = (np.sin(alt*np.pi/180))**2

    matrix = np.array([[sin_theta * 0.3, sin_theta * 0.16],
                       [sin_theta * 0.14, sin_theta * 0.4]], dtype=complex)

    quarter_wave_plate = np.array([[1, 0],
                                   [0, 1j]], dtype=complex)

    matrix1 = np.dot(quarter_wave_plate, matrix)

    conjugate_matrix = np.conjugate(matrix1)

    kronecker_product = np.kron(matrix1, conjugate_matrix)

    S_matrix = np.array([[0.5, 0.5, 0, 0],
                         [0, 0, 0.5, 0.5*1j],
                         [0, 0, 0.5, -0.5*1j],
                         [0.5, -0.5, 0, 0]], dtype=complex)

    result = np.dot(kronecker_product, S_matrix)

    return result
```

In the case of a Quarter wave plate, we will have the cross_mix, which will be a complex quantity. Each of the temparature maps (T_{xx}, T_{xy}, T_{yx} and T_{yy}) will be a complex in nature for which the real and imaginary parts are plotted separately.

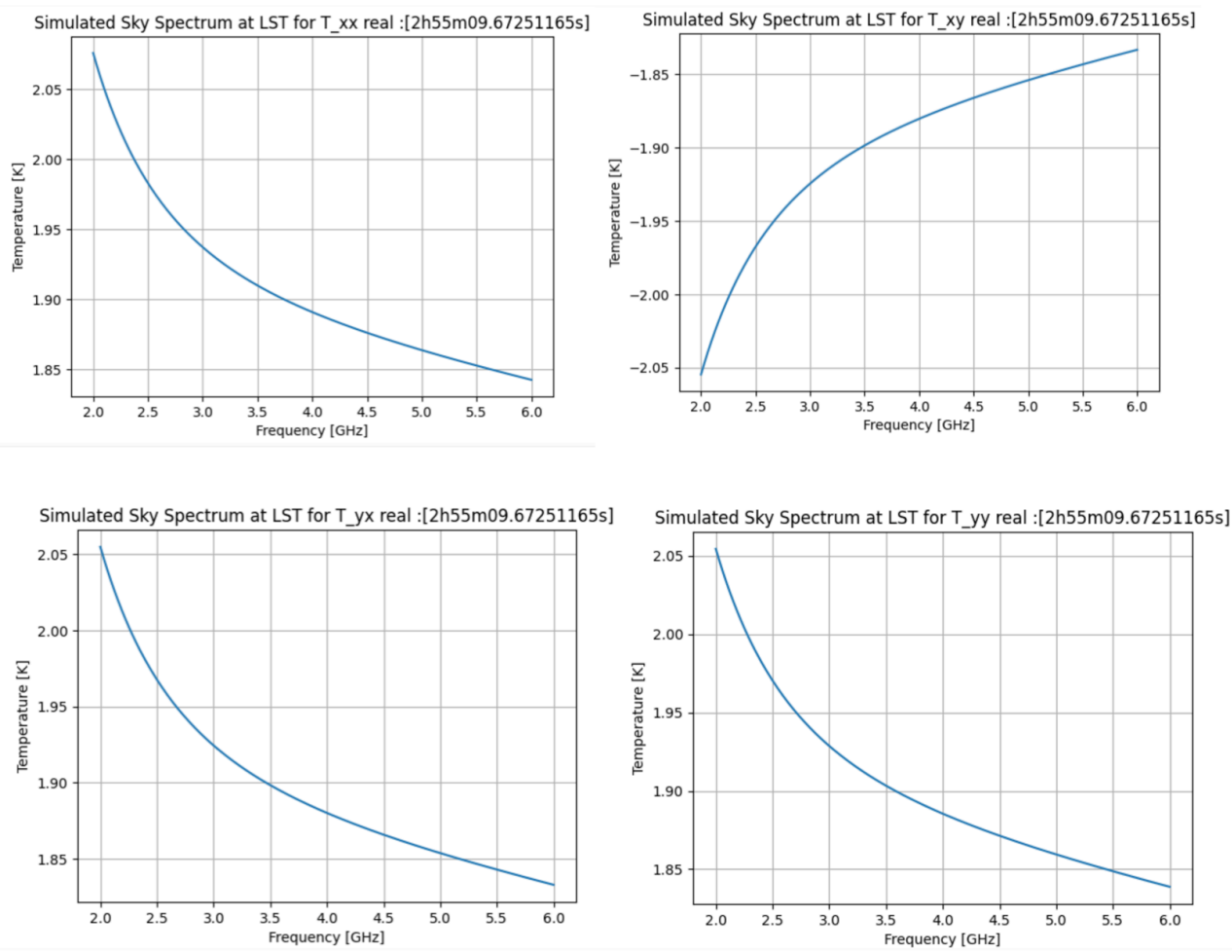


FIG 10: Real Spectra

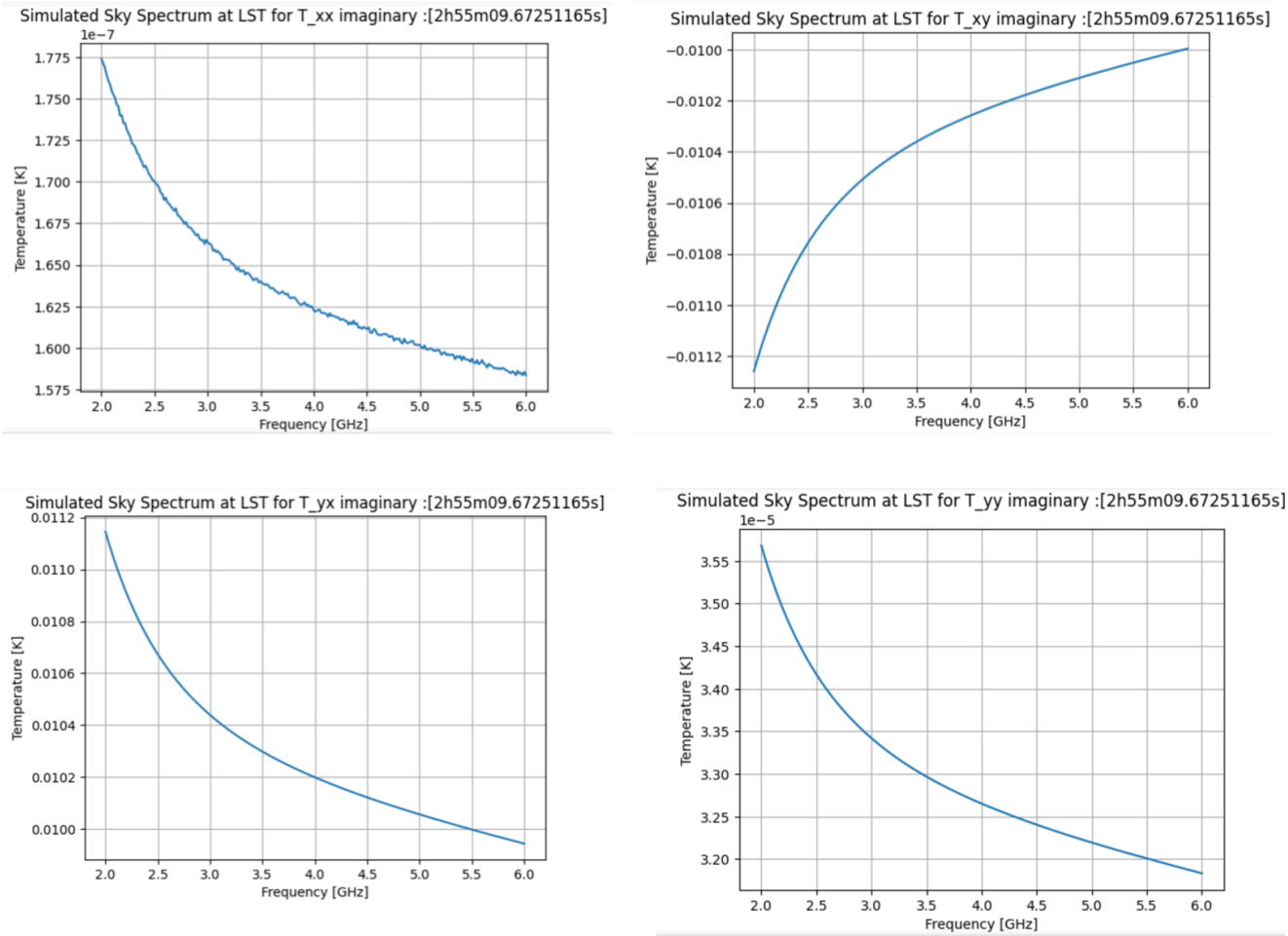


FIG 11: Imaginary Spectra

We also find the magnitude and phase for each of the case. Consider a complex number, a+ib, its magnitude is given by $\sqrt{a^2+b^2}$ and its phase is given by $tan^{-1}(\frac{b}{a})$. The graphs below are the magnitude and the phase for the specta(s) given above.

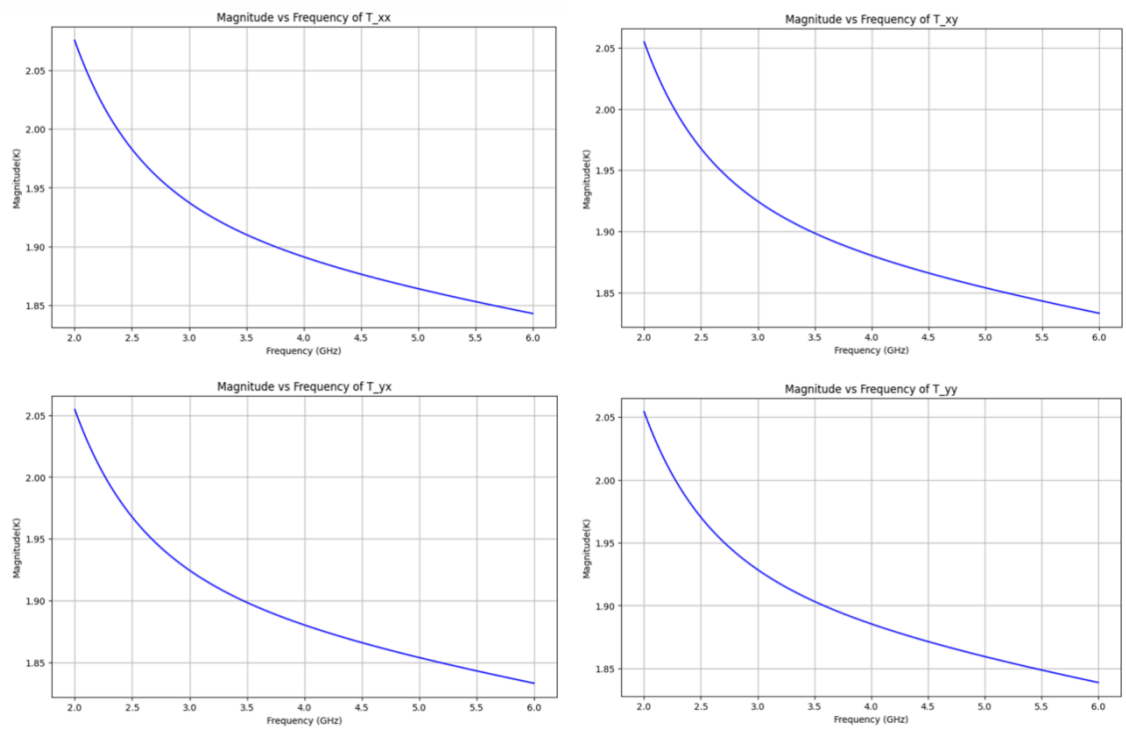


FIG 12: Magnitude vs Frequency

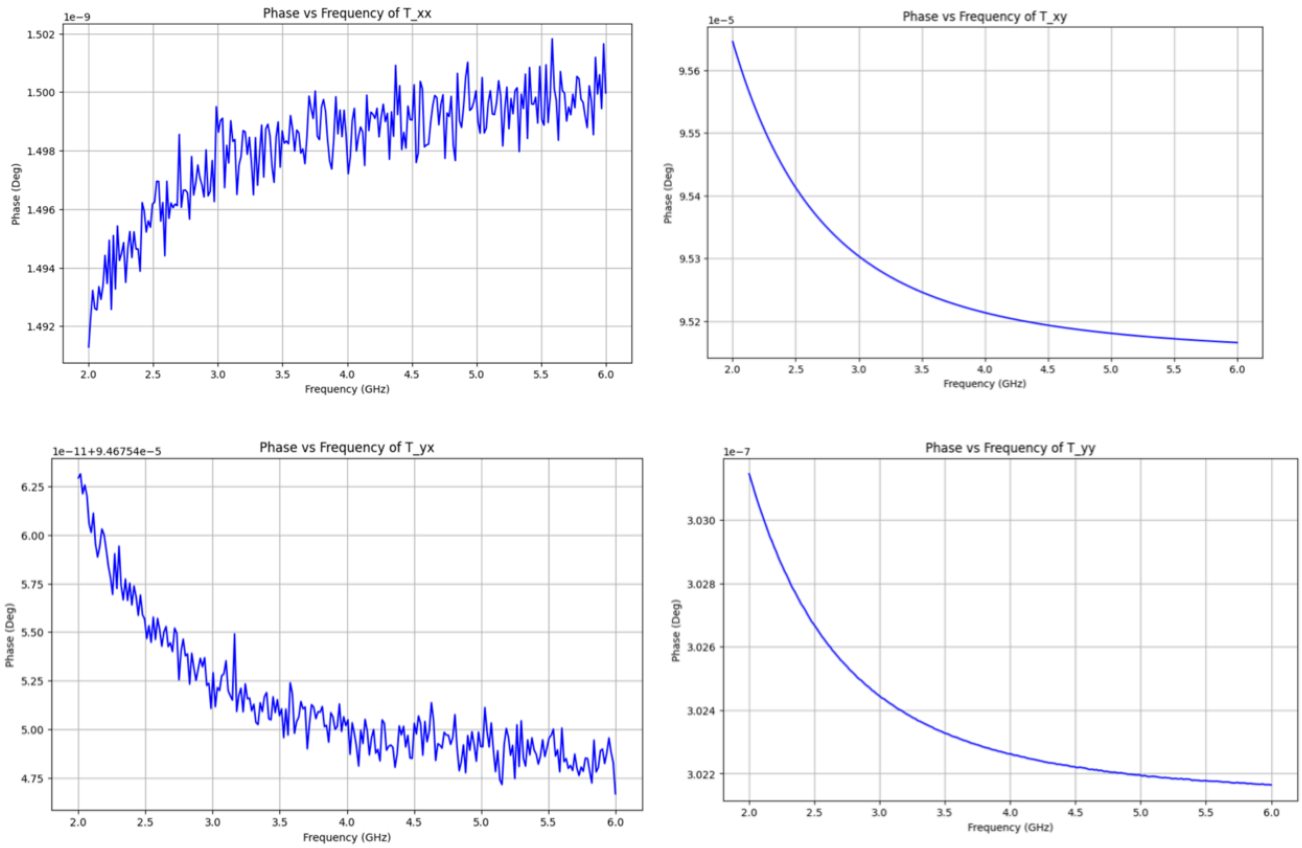


FIG 13: Phase vs Frequency

2.4. Rotating Quarter wave plate:

Now let us consider the case wherein we have a retarder plate rotated by an angle θ, from the initial pass axis. In this case, we'll have to consider the contribution of only the components of the electric field along the pass axis. Hence,

$E_0' = R(\theta)E_0 \qquad E_1' = R(\theta)E_1 \ldots 19$

where the rotation matrix is given by, $R(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \ldots 20$

Note on rotation matrix; $R(\theta)^{-1} = R(\theta)^T = R(-\theta)$.

Rotating E_1 by θ and inserting the identity matrix $I = R(\theta)R(\theta)^{-1}$, we get,

$E_1' = R(\theta)E_1 = R(\theta)JE_0 = R(\theta)JIE_0 = R(\theta)J(R(\theta)^{-1}R(\theta))E_0 = (R(\theta)JR(\theta)^{-1})(R(\theta)E_0)$

$=> E_1' = J'E_0' \ldots 21$

where $J' = R(\theta)JR(\theta)^{-1}$, which is the general expression for a Jones matrix to be rotated by an angle θ from its initial position.

We will put the above information to use. We already have that the Jones matrix for a horizontal linear polarizer is, $\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$. We rotate the horizontal linear polarizer by an angle θ.

$$J' = R(\theta)JR(\theta)^{-1} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} cos(\theta) & sin(\theta) \\ -sin(\theta) & cos(\theta) \end{bmatrix} = \begin{bmatrix} cos^2(\theta) & cos(\theta)sin(\theta) \\ cos(\theta)sin(\theta) & cos^2(\theta) \end{bmatrix}$$

...22

If we want to get the expression for +45^0 polarizer, then in the above expression we substitute $\theta = 45^0$, we get

$$J' = \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{bmatrix}$$

Consider the case where, $R(45^0) . Half\ wave\ plate\ . R(-45^0)$, this helps us switch the linear polarization state of the incoming electric field, i.e.,

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} <-> \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Also, $R(45^0) . Half\ wave\ plate\ . R(-45^0)$, this helps us switch the horizontal linear polarization state of the incoming electric field to right circularly polarized light and switches the verticle linear polarization state of the incoming electric field to left circularly polarized light.

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} <-> \begin{bmatrix} 1 \\ i \end{bmatrix} \qquad and \qquad \begin{bmatrix} 0 \\ 1 \end{bmatrix} <-> \begin{bmatrix} 1 \\ -i \end{bmatrix}.$$

If what a rotating retarder plate we replace, θ with ωt. Where ω is the angular frequency given by, $\frac{2\pi}{T}$, where "T" is the time period of one revolution (set to 60 seconds in our case). The above has been implemented into the mixing matrix as described below,

```python
def mixing_matrix2(frequencies, alt, az, timee):
    sin_theta = (np.sin(alt*np.pi/180))**2 #assuming alt is in degrees

    matrix = np.array([[sin_theta * 0.3, sin_theta * 0.16],
                       [sin_theta * 0.14, sin_theta * 0.4]], dtype=complex)

    half_wave_plate = np.array([[1, 0],
                                [0, 1j]], dtype=complex)

    T = 60 #in seconds, and defining "time" is seconds too

    rotation_matrix = np.array([[np.cos((2 * np.pi/T) * timee), -1 * np.sin((2 * np.pi/T) * timee)],
                                [np.sin((2 * np.pi/T) * timee), np.cos((2 * np.pi/T) * timee)]], dtype=complex)

    rotation_matrix_transpose = np.transpose(rotation_matrix)

    matrix1 = np.dot(rotation_matrix_transpose,matrix)
    matrix2 = np.dot(half_wave_plate, matrix1)
    matrix3 = np.dot(rotation_matrix, matrix2)

    conjugate_matrix = np.conjugate(matrix3)

    kronecker_product = np.kron(matrix3, conjugate_matrix)

    S_matrix = np.array([[0.5, 0.5, 0, 0],
                         [0, 0, 0.5, 0.5*1j],
                         [0, 0, 0.5, -0.5*1j],
                         [0.5, -0.5, 0, 0]], dtype=complex)

    result = np.dot(kronecker_product, S_matrix)

    return result
```

The spectras are recorded for a total time of two minutes in the interval of 5 seconds. This change is updated to the code as,

```python
timee = [ 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120]

for tim in timee:

    for h in range(4):

        t = 0
        while t < n_spectra:
            i = 0
            while i < 12288:  # Working on each pixel
                coords = hp.pixelfunc.pix2ang(32, i, nest = True, lonlat=True)  # Gives l and b coordinates for each pixel
                gc = SkyCoord(l=coords[0] * u.degree, b=coords[1] * u.degree, frame='galactic')
```

We now plot the Brightness temperature vs Time for a fixed frequency.

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

times = [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120]

file_paths_xx = [f'/Users/hshishir/Desktop/polarization_maps/rotating_halfwave1/spectrum_CRR_realT_xx_{tim}.txt' for tim in times]
file_paths_yy = [f'/Users/hshishir/Desktop/polarization_maps/rotating_halfwave1/spectrum_CRR_realT_yy_{tim}.txt' for tim in times]
file_paths_xy = [f'/Users/hshishir/Desktop/polarization_maps/rotating_halfwave1/spectrum_CRR_realT_xy_{tim}.txt' for tim in times]
file_paths_yx = [f'/Users/hshishir/Desktop/polarization_maps/rotating_halfwave1/spectrum_CRR_realT_yx_{tim}.txt' for tim in times]

channel11 = 251 #to set a particular channel (frequency between 2 to 6 GHz)

frequencies1_yy = [] spectra11_yy = [] frequencies1_xx = [] spectra11_xx = [] frequencies1_yx = [] spectra11_yx = []
frequencies1_xy = [] spectra11_xy = []

for file_path in file_paths_yy:
    data = np.loadtxt(file_path, dtype=complex)
    frequencies1_yy.append(data[channel11, 0].real)
    spectra11_yy.append(data[channel11, 1].real)

for file_path in file_paths_xx:
    data = np.loadtxt(file_path, dtype=complex)
    frequencies1_xx.append(data[channel11, 0].real)
    spectra11_xx.append(data[channel11, 1].real)

for file_path in file_paths_yx:
    data = np.loadtxt(file_path, dtype=complex)
    frequencies1_yx.append(data[channel11, 0].real)
    spectra11_yx.append(data[channel11, 1].real)

for file_path in file_paths_xy:
    data = np.loadtxt(file_path, dtype=complex)
    frequencies1_xy.append(data[channel11, 0].real)
    spectra11_xy.append(data[channel11, 1].real)

frequencies1_yy = np.array(frequencies1_yy)
spectra11_yy = np.array(spectra11_yy)
frequencies1_xx = np.array(frequencies1_xx)
spectra11_xx = np.array(spectra11_xx)
frequencies1_yx = np.array(frequencies1_yx)
spectra11_yx = np.array(spectra11_yx)
frequencies1_xy = np.array(frequencies1_xy)
spectra11_xy = np.array(spectra11_xy)
plt.figure(figsize=(10, 6))

plt.scatter(times, spectra11_{TT}, label='T_{} Data', color='green')
plt.plot(times, spectra11_xx, color='green')
plt.scatter(times, spectra11_yy, label='T_yy Data', color='blue')
plt.plot(times, spectra11_yy, color='blue')
plt.scatter(times, spectra11_xy, label='T_xy Data', color='red')
plt.plot(times, spectra11_xy, color='red')
plt.scatter(times, spectra11_yx, label='T_yx Data', color='black')
plt.plot(times, spectra11_yx, color='black')
```

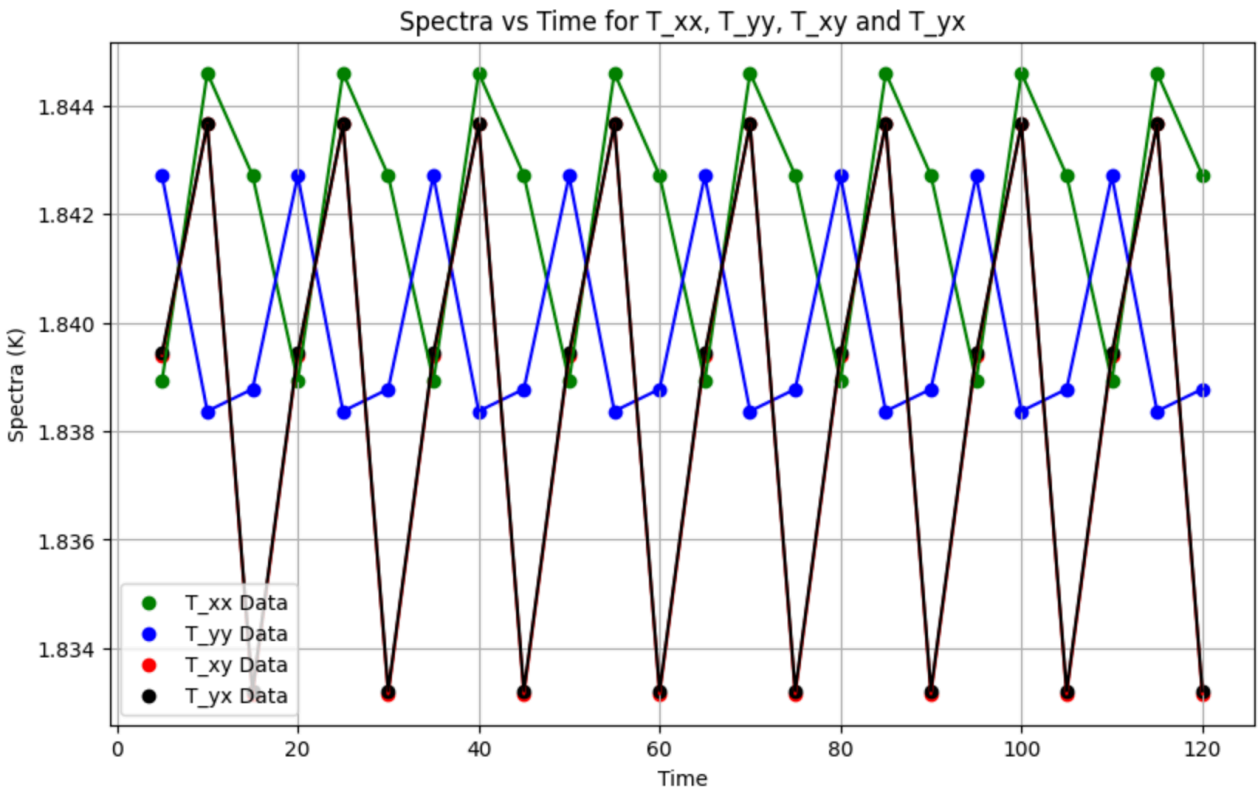


FIG 14: Brightness Temparature vs Time for a frequency near 2 GHz

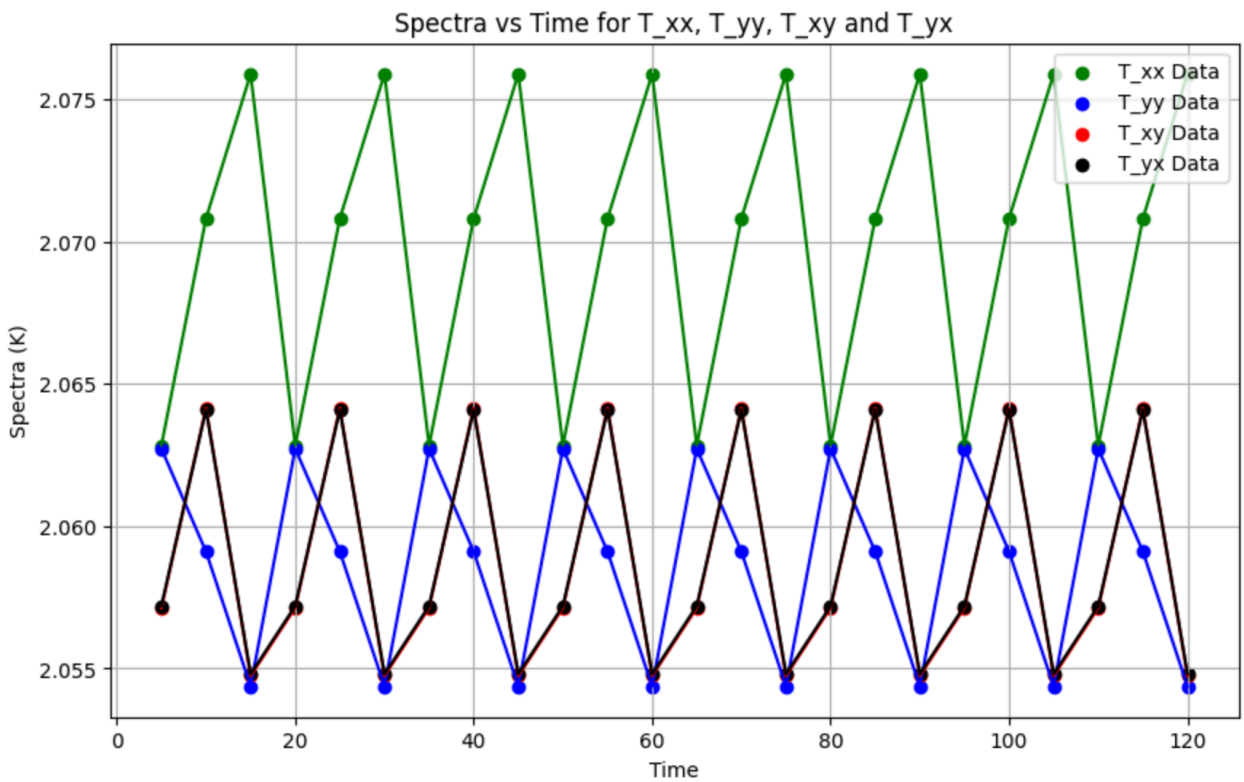


FIG 15: Brightness Temparature vs Time for a frequency near 6 GHz

3. MS Function:

3.1. Signal Extraction:

We will first have a breakdown Singnal Extraction section (Section 3) of Spinelli's paper. The Posterior is defined to be directly proportional to the Likelihood and Prior, given as,

$Posterior \propto Likelihood \times Prior$ ...1

Likelihood is defined as the probability distribution over the parameter before observing the data. *Prior* is the probability of the observed data given a specific set of parameter values (which is a function of the parameter for a fixed value). Ultimately, equation (1) can be written as,

$$p(\overrightarrow{\theta} \mid T(\nu_i)) = \frac{L(T(\nu_i) \mid \overrightarrow{\theta})p(\overrightarrow{\theta})}{p(T(\nu_i))} \ldots 2$$

where $\overrightarrow{\theta}$ represents the set of all parameters as a vector in the parameter space. The denominator is a normalization factor, wherein it is Likelihood times Prior integrated of the entire parameter space (i.e. the entire parameter space collapsed), given by,

$$p(T(\nu_i)) = \int_{\theta} L(T(\nu_i) \mid \overrightarrow{\theta})p(\overrightarrow{\theta})d\overrightarrow{\theta} \ldots 3$$

Here, the Likelihood function is chosen to be a gaussian,

$$L(T(\nu_i) \mid \overrightarrow{\theta}) = \frac{1}{\sqrt{2\pi\sigma_N^2(\nu_i)}} exp(-\frac{(T(\nu_i) - T_m(\nu_i, \overrightarrow{\theta}))^2}{2\sigma_N^2(\nu_i)}) \ldots 4$$

where $T(\nu_i)$ is data, σ_N is the radiometer noise, $\sigma_{xx,yy}^N = \frac{T_{xx,yy}(\nu)}{\sqrt{\Delta t \Delta \nu}}$, where Δt is integration time and $\Delta\nu$ is channel width and $T_m(\nu_i, \overrightarrow{\theta})$ is the model, $T_m(\nu_i, \overrightarrow{\theta}) = T_f + T_{21}$ where T_f is the foreground all-sky intensity temperature and three models of T_{21} have been discussed as per Low Frequency (LF) and High Frequency (HF) band,

1. $T_{21,G}^{LF}(\nu) = A_{21} exp(-\frac{(\nu - \nu_{21})^2}{2\sigma_{21}^2}) \ldots 5$

2. $T_{21,fg}^{LF}(\nu) = A_{21}(\frac{1 - e^{-\tau e^B}}{1 - e^{-\tau}})$, where $B = \frac{4(\nu - \nu_{21})^2}{\omega^2} log[-\frac{1}{\tau} log(\frac{1 + e^{-\tau}}{2})] \ldots 6$

3. $T_{21}^{HF}(z) = a_{21} x_{HI}(z) \sqrt{\frac{1+z}{10}}$, where $x_{HI}(z) = \frac{1}{2}[tan(\frac{z - z_r}{\Delta z}) + 1] \ldots 7$

We assume that our foregrounds are smooth and is modelled as an N-th order log polynomial,

$$log_{10}T_f(\nu) = \Sigma_{n=1}^{N}\ p_{n-1}[log_{10}\frac{\nu}{\nu_0}]^{n-1} \ldots 8$$

From the above, we can configure that the parameter space is composed of (assuming that we fit the foreground to N = 5),

$$\vec{\theta} = (\Delta z, z_r, p_0, p_1, p_2, p_3, p_4) \ldots 9$$

To extract the global 21-cm signal from the simulated spectra the `HIBAYES` code is used, a fully Bayesian framework where the posterior probability distribution is explored through the `MULTINEST` sampler using a Message Passing Interface (`MPI`)-enabled PYTHON wrapper.

3.2. Fitting Residuals:

Let us now consider the case where we obtain the all-sky spectrum without considering any polarization effects. Here, we assume that the foreground is *maximally smooth*. We define the *maximally smooth function* as follows,

$$f(x) = \Sigma_{i=0}^{n} p_i(x - x_0)^i \ldots 10$$

an n-th order polynomial. We require that all derivatives of order 2 and higher have no zero crossings within the domain of interest. To achieve this we implement the function,

$$\frac{d^m f(x)}{dx^m} = \Sigma_{i=0}^{n-m}\frac{(m+i)!}{i!}p_{m+i}(x - x_0)^i \ldots 11$$

for all m in the range 2, 3, 4, ..., (n – 1) and constraining the polynomial coefficients p_j so that there are no zero crossings within the domain for any of these functions.

We then fit our spectra to a maximally smooth function, as mentioned above by minimizing through χ^2 test. The spectra is now subtracted with the fitted maximally smooth function, to obtain the residuals. These residuals are now fitted to a simple sine curve of the form, $f(x) = Asin(2\pi\omega\nu + \phi)$. Ultimately we have three parameters Amplitude, frequency/ time-period and phase difference of the curve that have to be determined, i.e., A, ω and ϕ.

This is achieved using FFT. We use Fourier transform to find the ω. Which in the Fourier domain will just be a delta function at a particular value (as it is single-valued). After determining ω, we use `curve fit` from `scipy.optimize` to find the Amplitude and the phase for the residuals. Also to exclude degeneracy in the solution, we ensure that the phase difference is constrained as, $0 < \phi < 2\pi$. The following has been depicted in the code below, with relevant comments to help navigate through the code.

NOTE: Please go through the paper in the reference to understand better how to obtain the residuals. The code to to find the residuals in also available in the Jupyter Notebook.

```python
import numpy as np
import scipy.optimize
import matplotlib.pyplot as plt
import os

def fit_sin(tt, yy):
    '''Fit sine wave to the input time sequence, and return fitting parameters.'''
    tt = np.array(tt)  # Convert time/frequency array to numpy array for numerical operations
    yy = np.array(yy)  # Convert amplitude array to numpy array

    # Calculate the FFT of the input data to estimate the dominant frequency
    ff = np.fft.fftfreq(len(tt), (tt[1]-tt[0]))   # Create frequency bins for FFT, assuming uniform spacing
    Fyy = abs(np.fft.fft(yy))  # Compute the magnitude of the FFT to find the dominant frequency component

    # Initial guess for the fitting parameters based on FFT results
    guess_freq = abs(ff[np.argmax(Fyy[1:])+1])   # Guess frequency: the peak in FFT spectrum (ignoring the zero frequency)
    guess_amp = np.std(yy) * 2.**0.5  # Guess amplitude: estimated using the standard deviation of the data
    guess = np.array([guess_amp, 2.*np.pi*guess_freq, 0.]) # Initial guess array: amplitude, angular frequency, and phase

    # Define the sine function to fit to the data
    def sinfunc(t, A, w, p):
        return A * np.sin(w*t + p)  # A: amplitude, w: angular frequency, p: phase

    # Perform the curve fitting to optimize the sine wave parameters
    popt, pcov = scipy.optimize.curve_fit(sinfunc, tt, yy, p0=guess)
    A, w, p = popt  # Extract the optimized parameters: amplitude, angular frequency, and phase
    f = w/(2.*np.pi)  # Convert angular frequency back to regular frequency

    # Ensure the phase is within the range [0, 2  ]
    if p % (2 * np.pi) != 0:
        p = p % (2 * np.pi)
        p = np.mod(p, 2 * np.pi)

    # Define the fitted sine function using the optimized parameters
    fitfunc = lambda t: A * np.sin(w*t + p)

    # Return the fitting parameters and the fit function
    return {"amp": A, "omega": w, "phase": p, "freq": f, "period": 1./f,
            "fitfunc": fitfunc, "maxcov": np.max(pcov), "rawres": (guess, popt, pcov)}

directory = '/Users/hshishir/Desktop/mcmc_fit/sp2/'

# List of filenames in the directory that start with 'sp2_spectra' and end with '.txt'
files = [f for f in os.listdir(directory) if f.startswith('sp2_spectra') and f.endswith('.txt')]

# Iterate over each file, perform the fit, and plot the results
for file in files:
    # Construct the full file path and read the data from the file
    filepath = os.path.join(directory, file)
    data = np.loadtxt(filepath)

    y_noise = data  # Store the data into y_noise for processing

    # Fit the noisy sine wave to the data
    res = fit_sin(freqs, y_noise)

```

```python
    print(f"File: {file}")
    print("Amplitude=%(amp)s, Angular freq.=%(omega)s, phase=%(phase)s, Max. Cov.=%(maxcov)s" %
    res)

    plt.figure()
    plt.plot(freqs, y_noise, "ok", label="Noisy sine wave")
    plt.plot(freqs, res["fitfunc"](freqs), "r-", label="Fitted sine wave", linewidth=2)
    plt.xlabel('Frequency [GHz]')
    plt.ylabel('Amplitude')
    plt.title(f'Fit for {file}')
    plt.legend(loc="best")
    plt.grid()
    plt.show()
```

```python
directory = '/Users/hshishir/Desktop/mcmc_fit/sp2/'

files = [f for f in os.listdir(directory) if f.startswith('sp2_spectra') and f.endswith('.txt')]

for file in files:
    # Read the data from the file
    filepath = os.path.join(directory, file)
    data = np.loadtxt(filepath)

    y_noise = data

    res = fit_sin(freqs, y_noise)

    amplitude = res["amp"]
    omega = res["omega"] / (2 * np.pi)
    phase = res["phase"]
    phase_deg = phase * 57.3

    print(f"File: {file}")
    print("Amplitude =", amplitude)
    print("Omega =", omega)
    print("Phase =", phase)
    print("Phase in deg =", phase_deg)

    plt.figure(figsize=(10, 7))
    plt.plot(freqs, y_noise, label="Noisy sine wave", linewidth=2.0)
    plt.plot(freqs, amplitude * np.sin(2 * np.pi * freqs * omega + phase), "r-", label="Fitted
    sine wave", linewidth=2)
    plt.ylabel('Temperature (nK)', fontsize='xx-large', fontweight='bold')
    plt.xlabel('Frequency (GHz)', fontsize='xx-large', fontweight='bold')
    plt.xticks(fontsize=14, fontweight='bold')
    plt.yticks(fontsize=14, fontweight='bold')
    plt.grid(linewidth=1.5, linestyle='--')
    plt.legend(loc="best")
    plt.title(f'Fit for {file}')
    plt.show()

    fitted_data = amplitude * np.sin(2 * np.pi * freqs * omega + phase)
    output_filepath = os.path.join(directory, f"fit_{file}")
    np.savetxt(output_filepath, fitted_data)
    print(f"Fitted data saved to {output_filepath}")
```

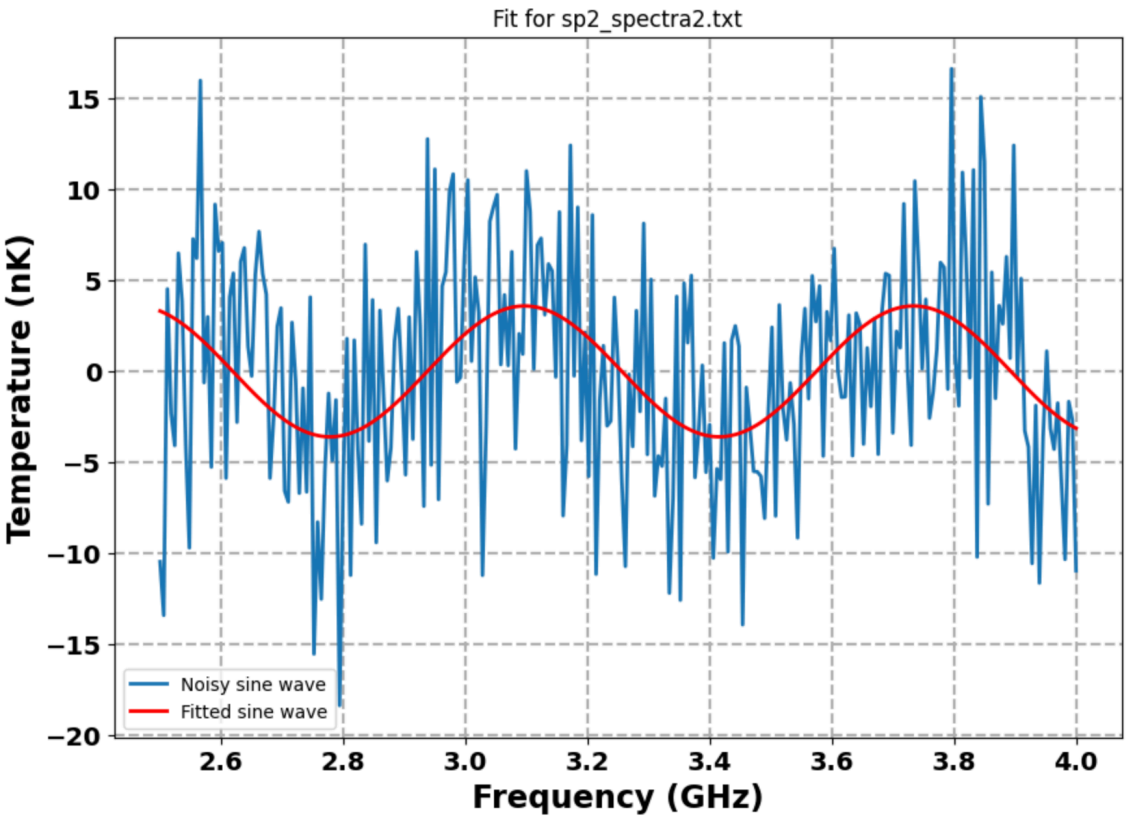


FIG 16: Residuals after fitting to a sine curve

We now plot each of the residuals alongside the fitted curves and print out the Amplitude, time-period and phase difference in radians and degrees.

We implement a model to fit a sine wave with intrinsic scatter to data from a set of spectra files. The `line_model` function defines the sine wave model, ensuring that the amplitude is non-negative and adjusting the phase accordingly. The `StraightLineModel` class is designed to compute the likelihood, prior, and posterior for a straight-line model with Gaussian errors, while the `StraightLineIntrinsicScatterModel` subclass handles the sine wave model with intrinsic scatter.

The code begins by defining initial parameters and then iterates over a set of spectra files, loading the data and associated uncertainties. For each spectrum, it creates an instance of `StraightLineIntrinsicScatterModel` using the frequency (freqs), data (y), and errors (y_err). The model parameters are then optimized using the minimize function from `scipy.optimize`, employing the "Powell" method. If the optimization is successful, the parameters are adjusted to ensure the amplitude is positive and the phase is within the correct range.

Finally, the code plots the best-fit sine wave along with the true sine wave for comparison, displaying the results for each spectrum. The plots include error bars on the data and show both the fitted sine wave and the true sine wave for visual comparison, helping to assess the accuracy of the fit.

```python
def line_model(pars, x):

    pars[0] * np.sin(2 * np.pi * x * pars[1] + pars[2])

    if pars[0] < 0:
        pars[0] = np.abs(pars[0])
        pars[2] = pars[2] - np.pi
        if pars[2] % (2 * np.pi) != 0:
            n = np.floor(pars[2] / (2 * np.pi))
            pars[2] = pars[2] - (n * 2 * np.pi)

    return pars[0] * np.sin(2 * np.pi * x * pars[1] + pars[2])

# Define the model class for the straight line with intrinsic scatter
class StraightLineModel:
    def __init__(self, x, y, y_err):
        self.x = np.asarray(x)
        self.y = np.asarray(y)
        self.y_err = np.asarray(y_err)

    def ln_likelihood(self, pars):
        N = len(self.y)
        dy = self.y - line_model(pars, self.x)
        ivar = 1 / self.y_err**2  # inverse-variance
        return -0.5 * (
            N * np.log(2 * np.pi)
            + np.sum(2 * np.log(self.y_err))
            + np.sum(dy**2 * ivar)
        )

    def ln_prior(self, pars):
        a, b, c = pars
        if a < 0 or a > 5:
            return -np.inf
        if b < 0 or b > 5:
            return -np.inf
        if c < -10 or c > 10:
            return -np.inf
        return 0.0

    def ln_posterior(self, pars):
        lnp = self.ln_prior(pars)
        if np.isinf(lnp):
            return lnp
        lnL = self.ln_likelihood(pars)
        lnprob = lnp + lnL
        if np.isnan(lnprob):
            return -np.inf
        return lnprob

    def __call__(self, pars):
        return self.ln_posterior(pars)

class StraightLineIntrinsicScatterModel(StraightLineModel):
    def ln_likelihood(self, pars):
        a, b, c = pars
        N = len(self.y)
        dy = self.y - line_model([a, b, c], self.x)
        return -0.5 * np.sum((dy / self.y_err) ** 2 + np.log(2 * np.pi * self.y_err**2))

# Define the initial parameters and true values
```

```python
x0 = [5.0, 1.5, 0.5]

def callbackF(pars):
    print(f"Current parameters: {pars}")

for i in range(n_spectra):
    # Load the y data and y_err data
    y = np.loadtxt(f'/Users/hshishir/Desktop/mcmc_fit/sp2/sp2_spectra{i}.txt')
    y_err = np.loadtxt(f'/Users/hshishir/Desktop/mcmc_fit/sigma_array/sigma_array_spectra{i}.txt'
    )

    # Create the scatter model
    scatter_model = StraightLineIntrinsicScatterModel(freqs, y, y_err)

    a_true = amplitude
    b_true = omega
    c_true = phase

    # Perform the optimization
    result_ml_scatter = minimize(
        lambda pars: -scatter_model.ln_likelihood(pars),
        x0=x0,
        method="Powell",
        callback=callbackF,
        options={'disp': True}
    )

    if not result_ml_scatter.success:
        print(f"Optimization failed: {result_ml_scatter.message}")

    if result_ml_scatter.x[0] < 0:
        result_ml_scatter.x[0] = np.abs(result_ml_scatter.x[0])
        result_ml_scatter.x[2] = result_ml_scatter.x[2] - np.pi
        if result_ml_scatter.x[2] % (2 * np.pi) != 0:
            nn = np.floor(result_ml_scatter.x[2] / (2 * np.pi))
            result_ml_scatter.x[2] = result_ml_scatter.x[2] - (nn * 2 * np.pi)

    print(f"Optimized parameters for spectrum {i}: {result_ml_scatter.x}")

    # Plot the results
    plt.figure()
    plt.errorbar(freqs, y, y_err, linestyle="none", marker="o", ecolor="#666666", label="data")

    x_grid = np.linspace(freqs.min() - 0.1, freqs.max() + 0.1, 128)
    plt.plot(
        x_grid,
        line_model(result_ml_scatter.x, x_grid),
        marker="",
        linestyle="-",
        label="best-fit line",
    )
    plt.plot(
        x_grid,
        line_model([a_true, b_true, c_true], x_grid),
        marker="",
        linestyle="dashed",
        label="true line",
    )

    plt.xlabel("$x$")
    plt.ylabel("$y$")
```

```
Current parameters: [-2.99459739  1.51059062  0.47879896]
Current parameters: [-3.10055006  1.51170577  0.47605757]
Optimization terminated successfully.
         Current function value: 805.408262
         Iterations: 2
         Function evaluations: 75
Optimized parameters for spectrum 0: [3.10055006 1.51170577 3.61765022]
```

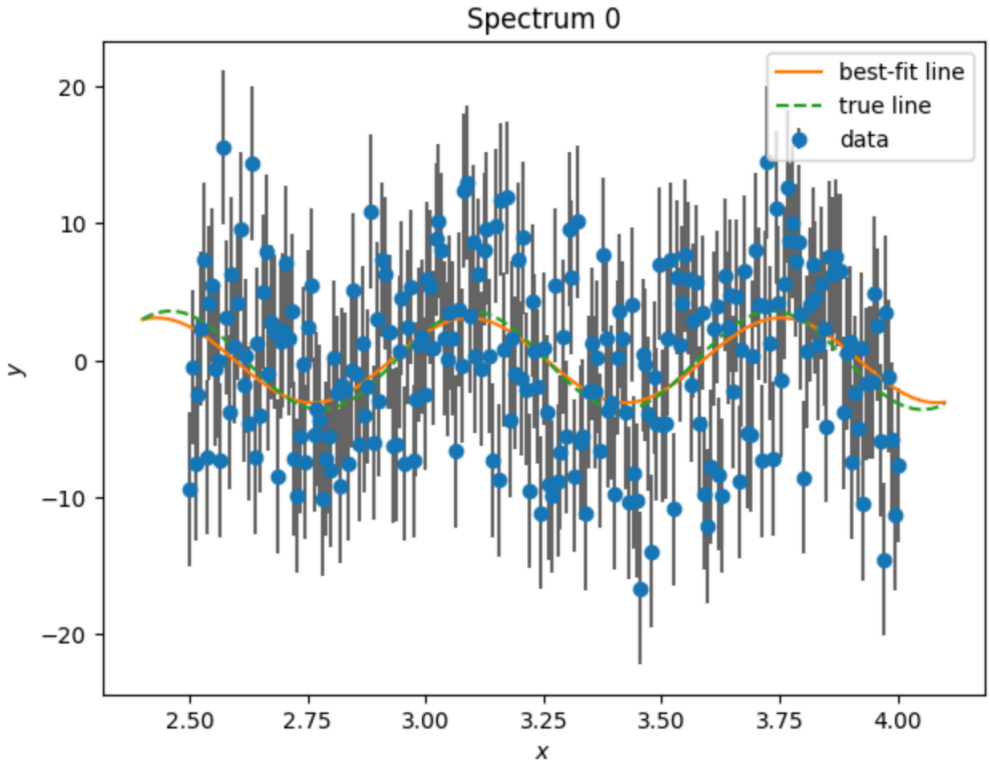


FIG 17: Residuals comparing the fitting to a sine curve and later minimizing it w.r.t log of likelihood

As depicted above in the code unlike Martha's paper, we work out everything is the log scale. Hence the Likelihood is given by,

$$log(Likelihood) = -0.5 \times (N \times \log(2\pi) + \sum 2 \times \log(\sigma) + \sum \frac{(y_{data} - y_{model})^2}{\sigma^2} \ldots 12$$

The prior is defined to be 0 for only a certain rage and $-\infty$ everywhere else. You can imagine it to be like a 1D square well potential which has a finite value potential in a certain range of the domain and infinity everywhere else. The prior can be 0 because we are taking the log of posterior ultimately, i.e.,

$$log(posterior) \propto log(Likelihood) + log(prior) \ldots 13$$

3.3. Bayesian Inference:

We now use Markov Chain Monte Carlo (MCMC) sampling to fit a sine wave model with intrinsic scatter to a series of spectral data sets. The code is broken down into the following main sections:

- Log-Posterior Probability Function (`lnprob`) —> The `lnprob` function calculates the log-posterior probability of the model parameters (`theta`) given the data (x, y, y_err). This function calls the `StraightLineIntrinsicScatterModel` to compute the log-posterior, which combines the likelihood and prior probability.
- MCMC Sampling Function (`main`) —> The `main` function runs the MCMC sampler:

- Burn-in phase: The initial phase to allow the walkers to explore the parameter space and settle into high-probability regions.

- Production phase: After resetting the sampler, the main sampling is conducted for `niter` iterations. The final positions, probabilities, and states of the walkers are returned.

- Plotting the Fit (`plotter`) —> The plotter function visualizes the fitted sine wave models:

- It plots the data points (x, y) and overlays sine wave models drawn from the MCMC chain (`samples`).

- This gives a visual representation of the fit and the spread of models sampled by MCMC.

- Sampling Walkers (`sample_walkers`) —> The `sample_walkers` function randomly selects a specified number (`nsamples`) of parameter sets from the MCMC chain. For each selected parameter set, the sine wave model is generated, and the spread (standard deviation) and median model are calculated across these sampled models.
- Plotting Highest Likelihood Model —> After MCMC sampling the best-fit parameters (`theta_max`) are extracted. A sine wave model is generated using these parameters and plotted alongside the data.
- Corner Plot (`corner.corner`) —> A corner plot is created to visualize the distribution of the sampled parameters. It displays histograms and correlation plots for the parameters a, b, and c, (Amplitude, time-period and phase difference) with the best-fit values (`theta_max`) indicated.
- Parameter Uncertainties —> The 16th, 50th (median), and 84th percentiles of the parameter distributions are calculated to provide estimates of the parameters and their uncertainties. These values are printed for each parameter.
- Plotting Posterior Spread —> Finally, the code samples walkers to generate a range of models, calculates the spread (uncertainty) in these models, and plots this as a filled region around the

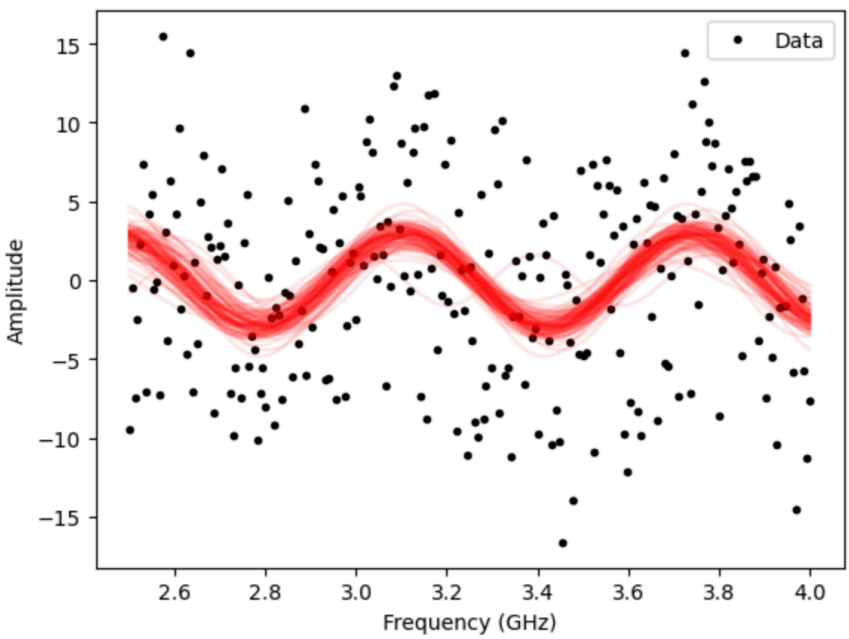
```python
import numpy as np
import emcee
import matplotlib.pyplot as plt
import corner

# Assuming 'StraightLineIntrinsicScatterModel' and 'line_model' are already defined

# Define the log-probability function that will be used by the MCMC sampler
def lnprob(theta, x, y, y_err):
    scatter_model = StraightLineIntrinsicScatterModel(x, y, y_err)
    return scatter_model.ln_posterior(theta)

# Main function to perform MCMC sampling
def main(p0, nwalkers, niter, ndim, lnprob, data):
    # Initialize the MCMC sampler
    sampler = emcee.EnsembleSampler(nwalkers, ndim, lnprob, args=data)

    # Run the burn-in phase to allow the sampler to converge
    print("Running burn-in...")
    p0, _, _ = sampler.run_mcmc(p0, 100)

    # Reset the sampler to discard the burn-in samples
    sampler.reset()

    # Run the production phase to generate samples
    print("Running production...")
    pos, prob, state = sampler.run_mcmc(p0, niter)

    return sampler, pos, prob, state

# Function to plot the MCMC results along with the data
def plotter(sampler, x, y):
    plt.ion()
    plt.plot(x, y, 'k.', label='Data')

    # Plot 100 random samples from the MCMC chain
    samples = sampler.flatchain
    for theta in samples[np.random.randint(len(samples), size=100)]:
        plt.plot(x, line_model(theta, x), color="r", alpha=0.1)

    plt.ticklabel_format(style='sci', axis='x', scilimits=(0, 0))
    plt.xlabel('Frequency (GHz)')
    plt.ylabel('Amplitude')
    plt.legend()
    plt.show()

# Function to sample walkers and calculate the median model and spread
def sample_walkers(nsamples, flattened_chain):
    models = []
    draw = np.floor(np.random.uniform(0, len(flattened_chain), size=nsamples)).astype(int)
    thetas = flattened_chain[draw]

    # Generate models using the sampled parameters
    for i in thetas:
        mod = line_model(i, freqs)
        models.append(mod)

    # Calculate the spread (standard deviation) and median model
    spread = np.std(models, axis=0)
    med_model = np.median(models, axis=0)

```

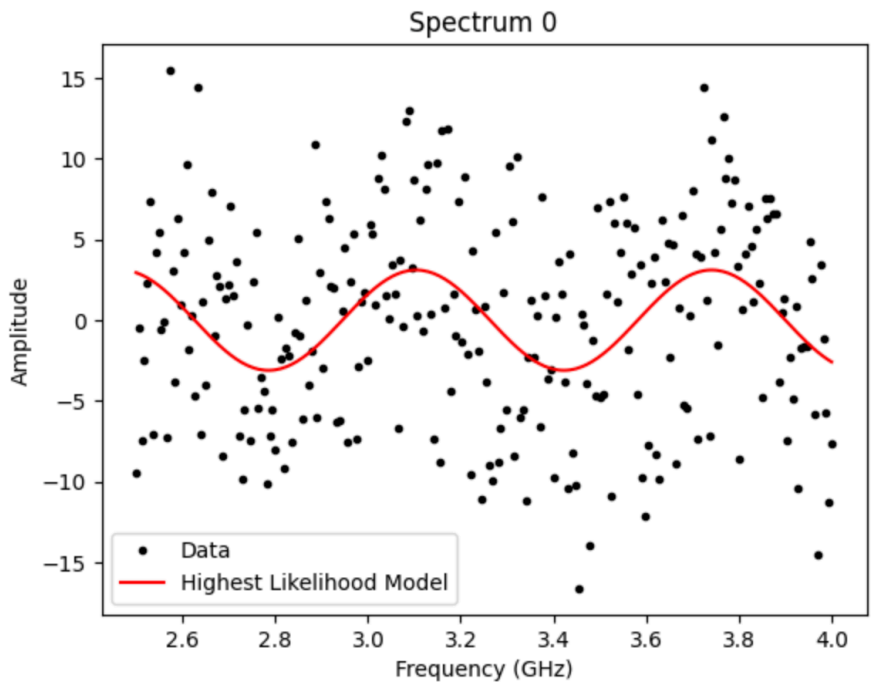
```python
    return med_model, spread

# Loop over all spectra
for i in range(n_spectra):
    # Load the y data and y_err data for the ith spectrum
    y = np.loadtxt(f'/Users/hshishir/Desktop/mcmc_fit/sp2/sp2_spectra{i}.txt')
    y_err = np.loadtxt(f'/Users/hshishir/Desktop/mcmc_fit/sigma_array/sigma_array_spectra{i}.txt')

    # Define MCMC parameters
    nwalkers = 250  # Number of walkers
    niter = 20000  # Number of iterations
    initial = np.array([1.0, 1.0, 8.0])  # Initial guess for parameters
    ndim = len(initial)  # Number of dimensions (parameters)

    # Initialize the walkers around the initial guess
    p0 = [np.array(initial) + 1e-7 * np.random.randn(ndim) for i in range(nwalkers)]

    # Perform MCMC sampling
    data = (freqs, y, y_err)
    sampler, pos, prob, state = main(p0, nwalkers, niter, ndim, lnprob, data)

    # Plot the MCMC results and the data
    plotter(sampler, freqs, y)

    # Find the parameters that maximize the likelihood
    samples = sampler.flatchain
    theta_max = samples[np.argmax(sampler.flatlnprobability)]

    # Adjust theta_max if necessary (to ensure proper phase wrapping)
    if theta_max[0] < 0:
        theta_max[0] = np.abs(theta_max[0])
        theta_max[2] = theta_max[2] - np.pi
        if theta_max[2] % (2 * np.pi) != 0:
            ee = np.floor(theta_max[2] / (2 * np.pi))
            theta_max[2] = theta_max[2] - (ee * 2 * np.pi)

    # Generate the best-fit model using the maximum likelihood parameters
    best_fit_model = line_model(theta_max, freqs)

    # Plot the data and the highest likelihood model
    plt.figure()
    plt.plot(freqs, y, 'k.', label='Data')
    plt.plot(freqs, best_fit_model, label='Highest Likelihood Model', color='r')
    plt.xlabel('Frequency (GHz)')
    plt.ylabel('Amplitude')
    plt.legend()
    plt.title(f'Spectrum {i}')
    plt.show()
    print(f'Theta max for spectrum {i}:', theta_max)

    # Create corner plot to visualize the parameter distributions
    labels = ['a', 'b', 'c']
    fig = corner.corner(samples, show_titles=True, labels=labels, plot_datapoints=True, quantiles=[0.16, 0.5, 0.84], truths=theta_max)
    plt.show()

    # Calculate and print parameter uncertainties (16th, 50th, and 84th percentiles)
    quantiles = np.percentile(samples, [16, 50, 84], axis=0)
    medians = quantiles[1]
    lower_errors = medians - quantiles[0]
```

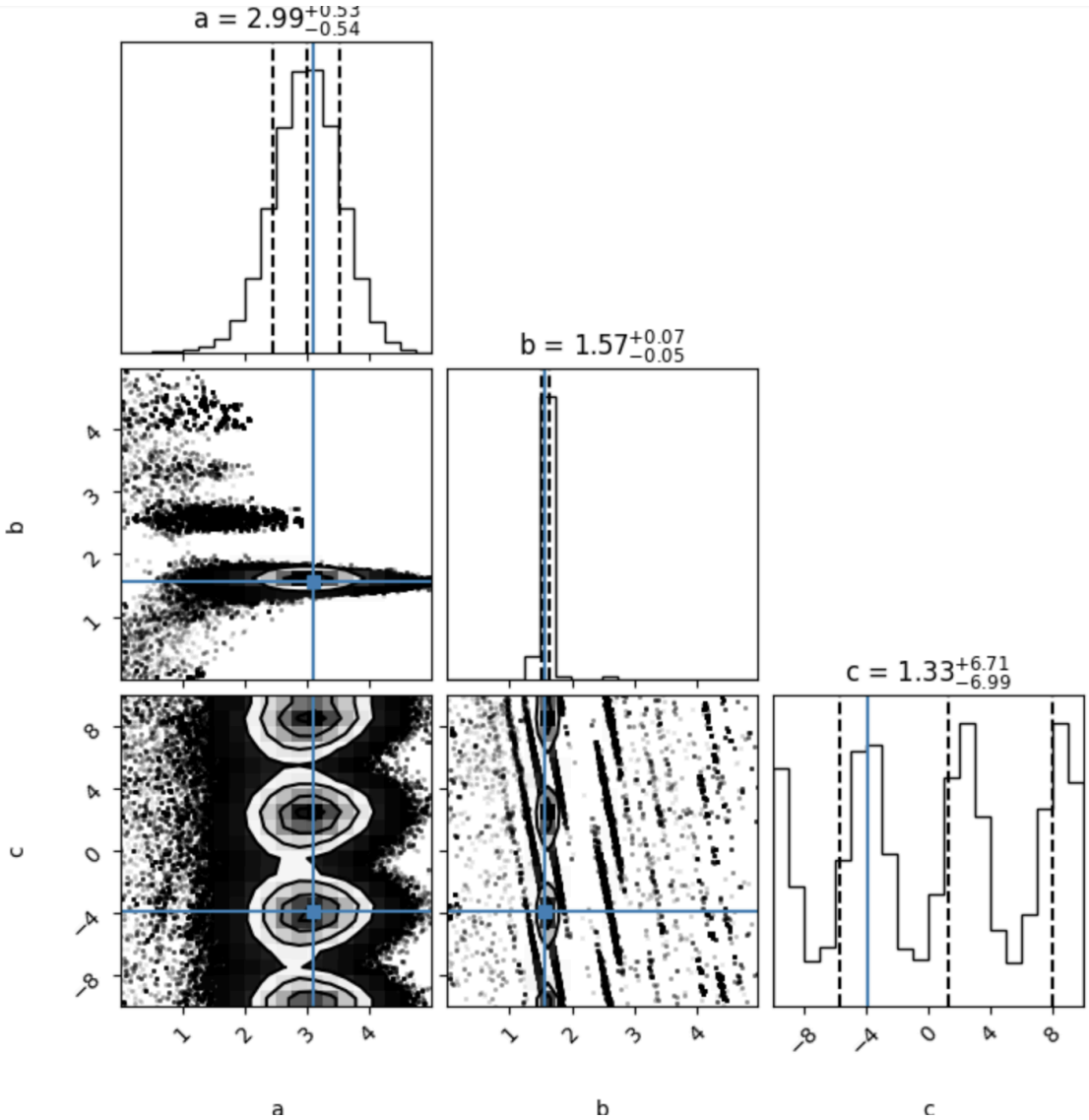
```python
    upper_errors = quantiles[2] - medians

    for j, label in enumerate(labels):
        print(f"{label}={medians[j]:.2f}^{{+{upper_errors[j]:.2f}}}_{{-{lower_errors[j]:.2f}}}")

    # Sample walkers and calculate the spread of the models
    med_model, spread = sample_walkers(100, samples)

    plt.plot(freqs, y, label='Data')
    plt.plot(freqs, best_fit_model, label='Highest Likelihood Model')
    plt.fill_between(freqs, med_model - spread, med_model + spread, color='red', alpha=1, label=r
    '$1\sigma$ Posterior Spread')
    plt.title(f'$1\sigma$ Posterior Spread Spectrum {i}')
    plt.legend()
    plt.show()
```

median model. This shows the uncertainty in the model predictions (1σ Posterior Spread) for each spectrum.

```
Running burn-in...
Running production...
```








```
a=2.99^{+0.53}_{-0.54}
b=1.57^{+0.07}_{-0.05}
c=1.33^{+6.71}_{-6.99}
```

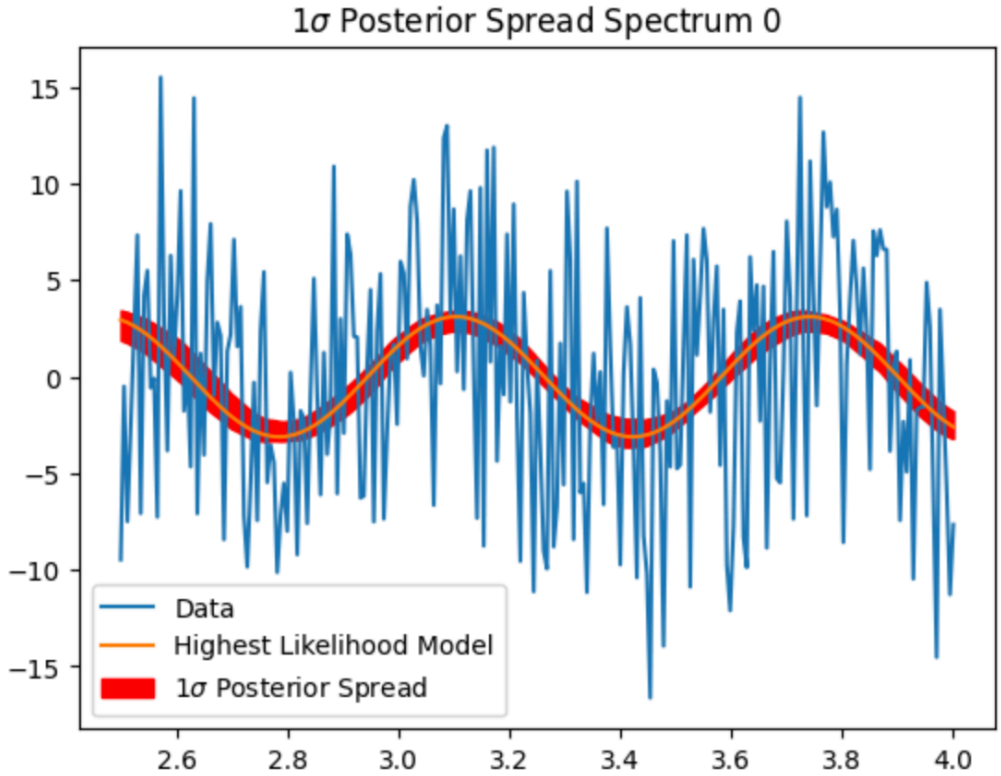


FIG 18: Bayesian Inferencing

The degeneracy in terms 'c' is because ϕ converges at every 2π, as all these solutions would fit the same curve.

NOTE: Here the residuals are of the order of nK.

3.4. Applying Bayesian Inference on the Half-wave / Quarter-wave plate:

Now that we have applied MCMC treatment on the T_{xx}, T_{xy}, T_{yz} and T_{zz} spectra separately. Look at the Jupyter notebook "APSERa_Ideal_Jones.ipynb", for the slight change in the code of MS Function. The results obtained are as follows. As an example, we'll perform the treatment on the T_{yz} element. We first look at the residue of T_{yz}.

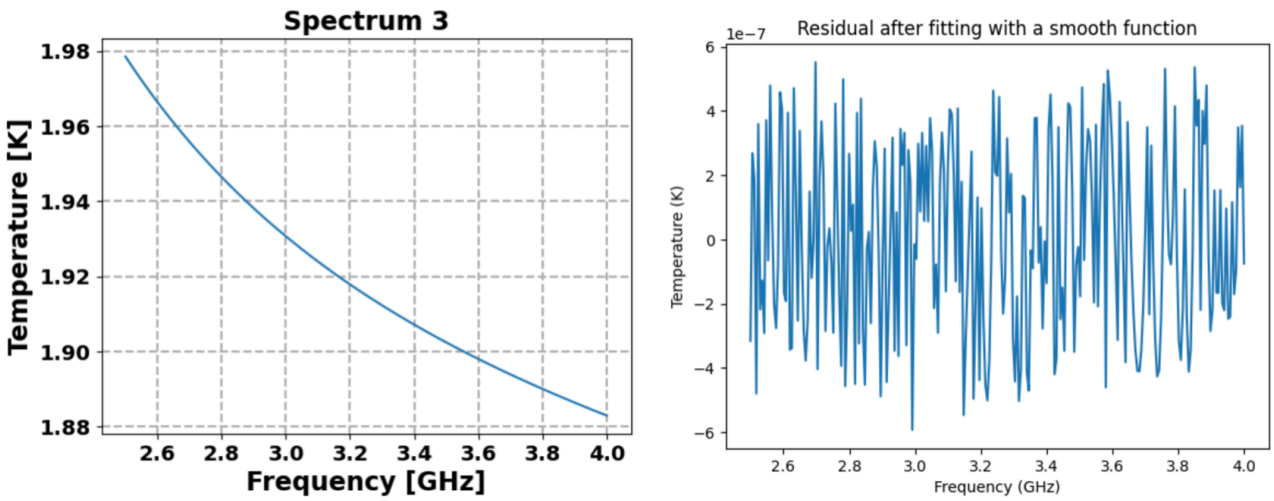


FIG 19: Tyz spectrum and it's corresponding residue

We now apply the MCMC treatment to the above residual by fitting it to the sine function. In order to fit the curve to a sine function previously, we tried using three ways: i) we used the inbuild SciPy's curve_fit function, ii) we used scipy.optimize to the given data after we had guessed the sine cure parameters crudely, and iii) we used scipy.optimize to the given data after we had guessed the sine cure parameters using Fast Fourier Transform.

We obtained our best results through the last method. Hence, we adapted the same here. But the results don't seem appealing. It can be seen from Fig. 19 that the residues, as viewed without any computational analysis, do not resemble the sine curve.

NOTE: Here, the residuals are of the order of 10^{-7}K.

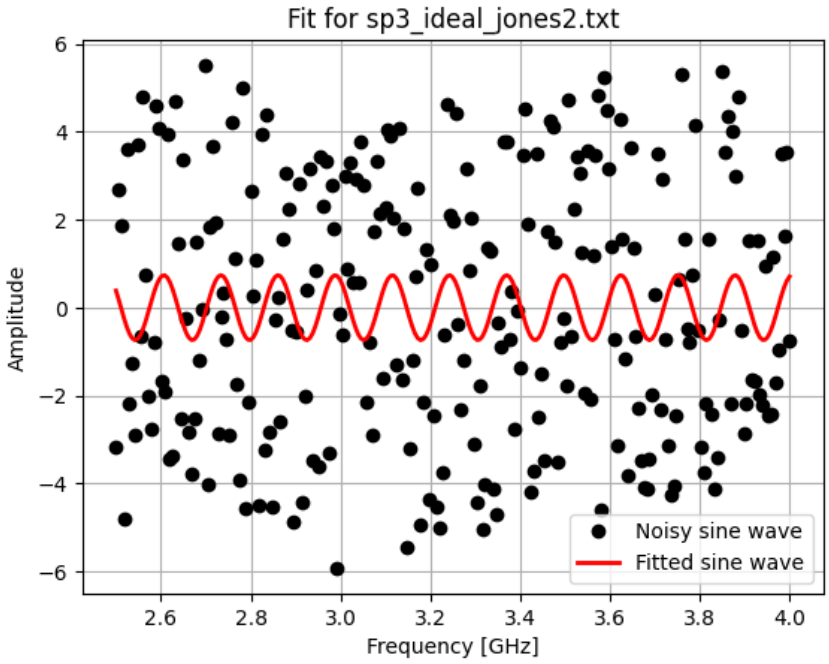
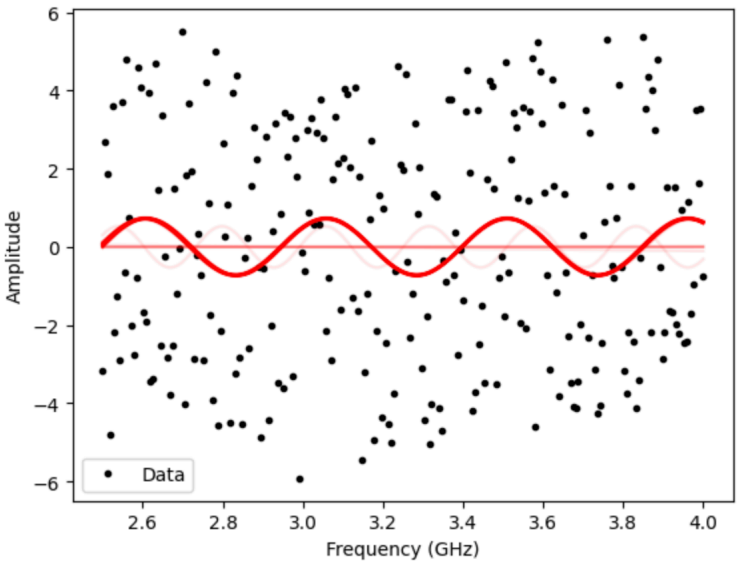
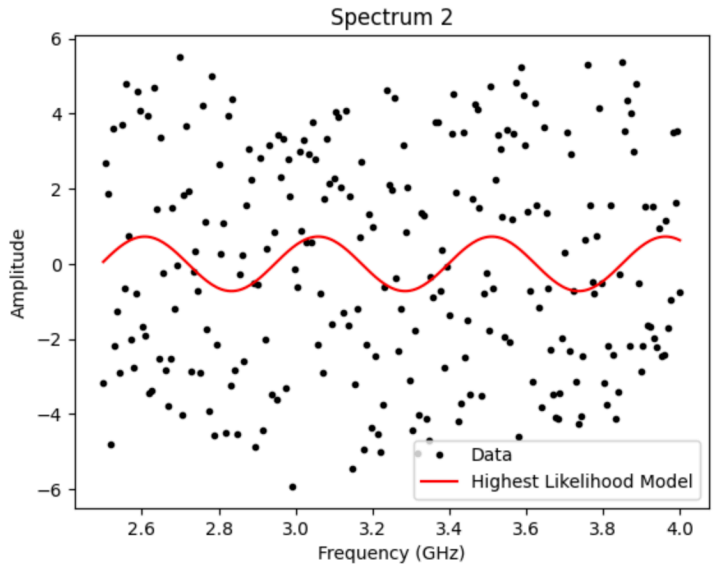


FIG 20: Tyz residue fit to sine curve using FFT

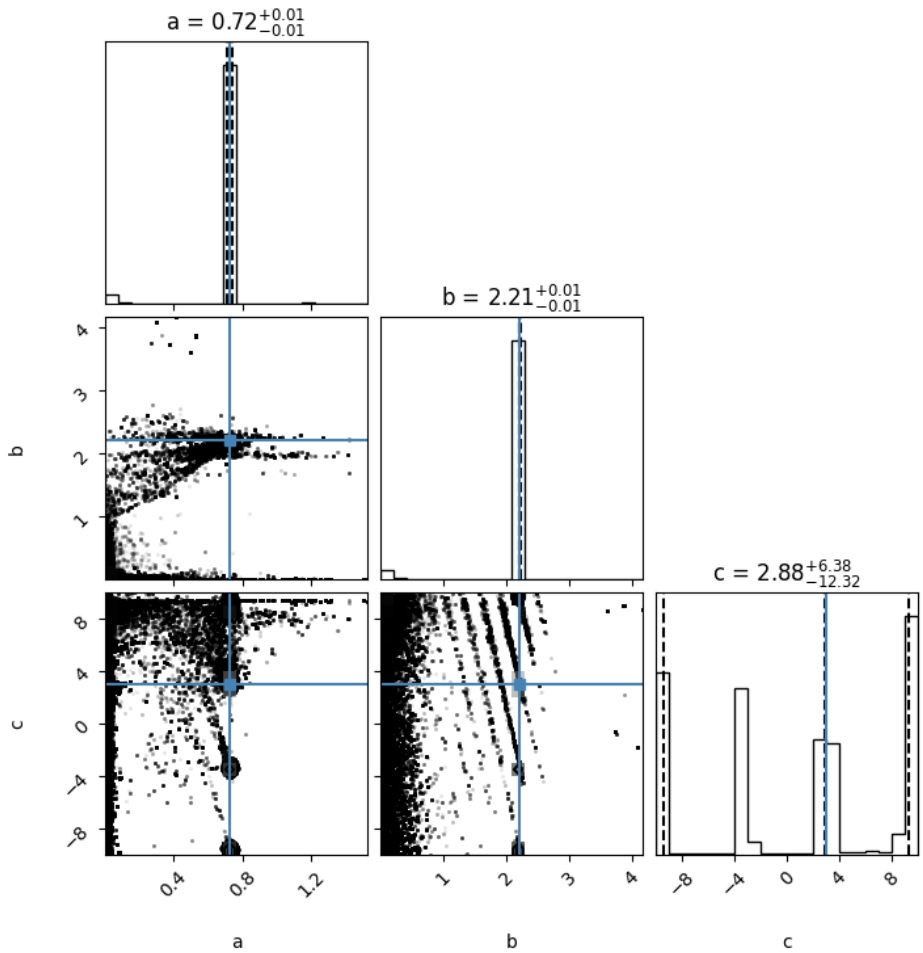
Clearly, we aren't able to fit the T_{yz} residue to the sine curve. This is also reflected in the MCMC analysis.

```
Running burn-in...
Running production...
```






```
Theta max for spectrum 2: [0.7255637600176088 2.2142102475967196 2.9935240805
```




```
a=0.72^{+0.01}_{-0.01}
b=2.21^{+0.01}_{-0.01}
c=2.88^{+6.38}_{-12.32}
```

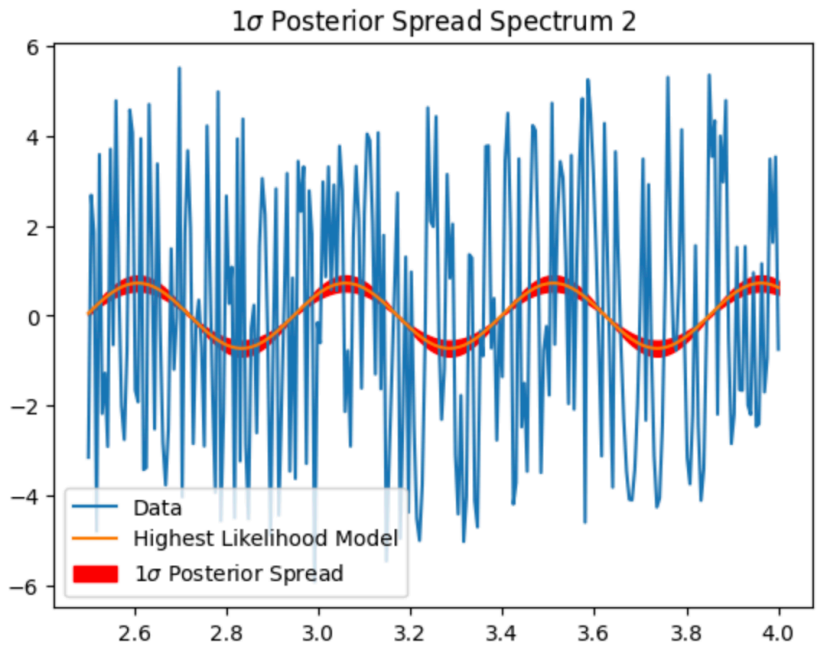


FIG 21: Tyz residue fit to sine curve using FFT

4. Links:

A. References Used for the Project:

1. https://academic.oup.com/mnras/article/489/3/4007/5558247
2. https://iopscience.iop.org/article/10.1088/0004-637X/767/2/150/meta
3. https://journals.aps.org/prd/abstract/10.1103/PhysRevD.89.123002
4. https://apc.u-paris.fr/~kaplan/Matrixpol/matrixpol.html
5. [**IMPORATANT**] On the contamination of the global 21 cm signal from polarized foregrounds by Marta Spinelli et. al. : https://arxiv.org/pdf/1908.05303
6. [**IMPORATANT**] On the Detection of Spectral Ripples from the Recombination Epoch by Mayuri Sathyanarayana Rao, Ravi Subrahmanyan, N Udaya Shankar, and Jens Chluba : https://iopscience.iop.org/article/10.1088/0004-637X/810/1/3/meta

B. Important Packages:

1. Healpy: https://healpy.readthedocs.io/en/latest/tutorial.html
2. Galprop: https://galprop.stanford.edu/code.php
3. Pygsm / Pygdsm:
   - [Pygdsm on GitHub] https://github.com/telegraphic/pygdsm
   - GSM2008: A model of diffuse Galactic radio emission from 10 MHz to 100 GHz, Oliveira-Costa et. al., (2008)
   - GSM2016: An improved model of diffuse galactic radio emission from 10 MHz to 5 THz, Zheng et. al., (2016)
   - LFSS: The LWA1 Low Frequency Sky Survey (10-408 MHz) Dowell et. al. (2017)
   - Haslam: A frequency-scaled model using a spectral index, based on the Haslam 408 MHz all-sky map
4. Pysm / Pysm3: [Pysm on GitHub](https://github.com/galsci/pysm)

C. Spherical Harmonic Basis and Power Spectrum:

1. Spherical Harmonic Basis and Power Spectrum Analysis: (https://iopscience.iop.org/article/10.3847/1538-4357/833/2/242#:~:text=In%20an%20intensity%20mapping%20survey,thus%20redshift)%20of%20the%20line
2. https://arxiv.org/abs/1005.0555

D. Bayesian Inference:

1. MCMC Tutorial and Bayesian Inference Overview: https://prappleizer.github.io/Tutorials/MCMC/MCMC_Tutorial.html#:~:text=MCMCs%20are%20a%20class%20of,%2C%20most%20simply%2C%20a%20sampler

2. Fitting a Line using Bayesian Methods: https://adrian.pw/blog/fitting-a-line/
3. Bilby: https://lscsoft.docs.ligo.org/bilby/code-overview.html

E. Essentials of Radio Astronomy:

1. Essentials of Radio Astronomy: https://www.cv.nrao.edu/~sransom/web/Ch2.html
2. Interferometry Visualized: https://public.nrao.edu/interferometry-explained/

F. Books:

1. A First Course in Fourier Anaysis by David W. Kammler
2. Tools of Radio Astronomy by Kristen Rohlfs and Thomas L. Wilson
3. Optics by Ajoy Ghatak
4. Numerical Methods of Statistics by John F. Monahan

G. Useful Wikipedia links:

1. https://en.wikipedia.org/wiki/Jansky
2. https://en.wikipedia.org/wiki/Spectral_index
3. https://en.wikipedia.org/wiki/Stokes_parameters
4. https://en.wikipedia.org/wiki/Spherical_harmonics

5. Jupyter Notebook Content:

1. **APSERa_Ideal_Jones.ipynb:** Spectra generated by a Jones matrix with non-diagonal terms zero (along with quarter wave/half-wave plate) + MS_fit Function + MCMC analysis
2. **APSERa_Non_Ideal_Jones.ipynb:** Spectra generated by a Jones matrix with non-diagonal terms as non-zero (along with quarter wave/half-wave plate); (Don't consider the "MS_fit Function" part)
3. **Apsera_modifiedbeam_dipole1simulation111-3.ipynb:** A working version of the orginal code for my system.
4. **APSERa_Rotating_Quarterwave_Plate.ipynb:** Spectra generated by a Jones matrix with along with quarter wave in which the θ is taken as ωt, and we plot Stokes parameters as a function of time
5. **Unpolarized_mcmc_final.ipynb:** We incorporate MCMC to the original code

H SHISHIR

Understanding Magnetic Field in disks of Spiral Galaxies

H Shishir[1, *]

[1] *School of Physical Sciences, National Institute of Science Education and Research, HBNI, Jatni, Khordha, Odisha-752050, India*

(Dated: August 7, 2023)

Abstract. In order to determine the configuration of the magnetic field of a spiral galaxy we need to look at the observables which we can relate to the magnetic field. This project aims at the review of the methods used in two papers: D. Sokoloff et al. (1992) which analyses the galaxies IC 342 and M81 and A. Fletcher et al. (2003) which analyses the galaxies M31. The position angles and the Faraday rotation measure provide to be appropriate observables which can be fitted to obtain the magnetic field parameters. The segment will include the discussion of the models used in each of the papers, also including a segment to emphasise the statistical measures to fit the data. These observables are additionally studied under varied cases which involves the inclination of the galaxy and the considering the pitch of the magnetic field lines. It is know that the magnetic field can have multiple modes. But here, the discussion is restricted to only the axisymmetric and bisymmetric fields by truncating the higher order coefficients of Faraday rotation measure and position angles.

Key words. spiral galaxy - IC 342 - M81 - Faraday rotation - axisymmetric field - bisymmetric field - inclination - pitch

1. Introduction

In the first paper of D. Sokoloff et al., we discuss analysing the regular magnetic field of spiral galaxies mainly by fitting the data of the Fourier series of Faraday rotation measure and position angles of spiral galaxies. We discuss and contrast two approaches to interpreting observations of spiral galaxies made using two frequencies of polarisation in order to determine the global structure of the large-scale magnetic field parallel and perpendicular to the disk. As suggested here, polarisation data can be understood directly in terms of position angles or indirectly in terms of Faraday rotation measures. The observations of the galaxies IC 342 and M81 are conducted using both techniques, which can be looked into. The latter technique, as it stands, works well for galaxies with either a low or high inclination. In cases of moderate inclination, the earlier method is still preferred even though it typically produces less precise results.

In the second paper, of A. Fletcher et al., we discuss the regular magnetic field of M31 using the observations of the radio polarisations, which are a function of the magnetic field at this time. From radio polarisation studies at the wavelengths $\lambda\lambda$6, 11, and 20 cm, the configuration of the normal magnetic field in M 31 is inferred. This is revealed by fitting the azimuthal distribution of reported polarisation angles. It is possible to pinpoint the primary cause of wavelength-dependent depolarisation through a thorough examination of the observed depolarisation. We do not delve into discussing the depolarisation and finding the magnetic field parameters in this case.

2. Review of D. Sokoloff et al. paper

2.1. The model

The first method of estimation of the magnetic field of spiral galaxies is the one where we analyse the polarisation pattern primarily using the Faraday rotation measure. We consider the azimuthal variations of Faraday rotation measure as RM. The following relations show the relation between RM and the position angle. The position angles at wavelengths λ_a and λ_b are given by ψ_a and ψ_b, respectively. The relation between the RM and the position angles at different wavelengths is given by:

$$RM = \frac{\psi_a - \psi_b}{\lambda_a - \lambda_b} \tag{1}$$

Now the intrinsic polarisation angle ψ_0, which we know holds information of the intrinsic magnetic field, is related to RM by:

$$\psi_o = \psi_a - RM\,.\,\lambda_a^2 \tag{2a}$$

$$\psi_o = \psi_b - RM\,.\,\lambda_b^2 \tag{2b}$$

In general, we can see from (2) that RM and ψ_0 are anti-correlated. We have two linear combinations of RM and ψ_0, which can be found out by varying ψ_a and ψ_b. If we are to consider RM measurements alone, neglecting ψ_0 , which might simplify the analysis, but half of the information of the intrinsic magnetic field will be lost.

RM can also be expressed as a Fourier series expansion with azimuthal angle θ measured in the galactic plane and is given by:

$$RM = r_0 + r_1 cos\theta + r_2 sin\theta + r_3 cos2\theta + r_4 sin2\theta + \ldots, \tag{3}$$

where coefficients r_1 and r_2 are related to the axisymmetric field and r_0, r_3 and r_4 correspond to the bisymmetric field and the foreground Faraday rotation. In order to find the

* shishir.h@niser.ac.in

parameters r_n , we can fit the data of RM against the azimuthal angle θ. In order to determine the reliability of the fit, we resort to χ^2 test and the Fisher criteria, which are discussed in Sect. 2.3.

2.2 Relating the magnetic field parameter to the model

The second method of estimation of the magnetic field of spiral galaxies is the one where we analyse the polarisation pattern primarily using the position angle. Position angles $\psi_a(\theta)$ and $\psi_b(\theta)$ like RM can also be expressed as a Fourier series expansion with azimuthal angle θ measured in the galactic plane, and is given by:

$$\psi_a(\theta) = a_0 + a_1 cos\theta + a_2 sin\theta + a_3 cos2\theta + a_4 sin2\theta + \ldots, \quad (4a)$$

$$\psi_b(\theta) = b_0 + b_1 cos\theta + b_2 sin\theta + b_3 cos2\theta + b_4 sin2\theta + \ldots, \quad (4b)$$

Where ψ_a and ψ_b are values of the position angle at wavelengths ψ_a and ψ_b, respectively. In order to find the parameters a_n and b_n we can fit the data of RM against azimuthal angle θ, which are connected to the intrinsic magnetic field of a given galaxy.

The RM in (2) can also be given by $RM = RM(\theta) + R_{fg}$ where $RM(\theta)$ is the intrinsic Faraday rotation measure and RM_{fg} is the foreground rotation measure produced in the space between the galaxy and the observer. In A. Fletcher et al. RM_{fg} is the foreground rotation measure of the Milky Way. Hence, (2) can be rewritten as:

$$\psi_a(\theta) = RM(\theta).\lambda_a^2 + RM_{fg}.\lambda_a^2 + \psi_0(\theta), \quad (5a)$$

$$\psi_b(\theta) = RM(\theta).\lambda_b^2 + RM_{fg}.\lambda_b^2 + \psi_0(\theta), \quad (5b)$$

where $\psi_0(\theta)$ is the intrinsic portion angle at the azimuthal angle θ in the considered galaxy.

We consider i to be the angle at which a galactic disc whose rotation axis is inclined to the line of sight, B be the regular field strength and the pitch angle of the magnetic lines is given by p. It is theoretically and observationally established that the regular magnetic field in the spiral galaxy lies predominantly in the disc plane. Hence, we usually neglect the B_z, and $B_r = Bsinp$ and $B_\theta = Bcosp$. If we include B_z, we get the magnetic field vector $\boldsymbol{B}$.

We will now look at the expressions $\psi_0(\theta)$ for different cases. First, the magnetic field is purely horizontal field is given by:

$$\psi_0(\theta) = arctan\frac{1}{cosi}\frac{tan(\theta - p) - tan\theta cos^2 i}{1 + tan\theta tan(\theta - p)} \quad (6)$$

We note that equation (6) is independent of B and is a purely geometric result. The equation (6) has been plotted in Fig. 1. The pitch has been kept constant at p=-20^0, and the angle of inclination is varied as i=20^0, 50^0 and 80^0. We can note that the frequency of the plot remains the same where as the amplitude of the plot increases as we go from 20^0 to 80^0. Second, if we include the veritable magnetic field B_z, we get:

$$\psi_0(\theta) = arctan\left(\frac{1}{2}\frac{B[sin(2\theta - p) - (1 + cos^2 i)sinp]}{Bcosicosp + B_z sinicos\theta} - \frac{1}{2}\frac{B_z sin2isin\theta}{Bcosicosp + B_z sinicos\theta}\right) \quad (7)$$

Fig. 2 shows the plot of equation (7) by keeping p and i and varying the ratio of $\frac{B_z}{B}$. It can be noticed that if the ratio is less than or equal to 1, then the curve is smooth and almost sinusoidal. Thirdly, if it's a purely vertical magnetic field, then equation (7) gives us:

$$\psi_0(\theta) = -arctan(cosi.tan\theta)$$

$$\psi_0(\theta) = -\theta' \qquad \text{for } 0 < \theta < 90^0,$$

$$\psi_0(\theta) = -\theta' + 180^0 \qquad \text{for } 270^0 < \theta < 360^0, \quad (8)$$

here θ' is the azimuthal angle measured in the sky. It can be seen that (8) is close to a pair-wise linear function of θ with a period of 180^0. (7) is also discontinuous.

The line-of-sight component of the magnetic field $\boldsymbol{B}$ is represented by an axisymmetric and bisymmetric components of the strengths B_0 and $B_1 cos(\theta - \beta)$, respectively. Once we determine **B,** we can deduce the parameters for the regular magnetic field, i.e., the coefficients $a_1, b_1, r_1, r_2, \ldots$. The line-of-sight component of the magnetic field is given by:

$$B_{||} = -|B|cos(\theta - p)sini$$
$$= -[B_0 cos(\theta - p_0) + B_1 cos(\theta - \beta)cos(\theta - p_1)]sini \quad (9)$$

where p_o and p_i are the pitch angles of magnetic lines of the axisymmetric and bisymmetric field, respectively, the phase β is defined as the value of θ at which the bisymmetric field has a maximal strength and is directed along the axisymmetric one, and i is the angle between the angular velocity vector and the line of sight, known also as the inclination angle.

The physical parameters of the magnetic field are related to the coefficients, i.e., $a_n, b_n and r_n$, for **low inclination,** the expressions are:

$$a_1 = \lambda_a^2 RM_{fg} - \frac{1}{2}\lambda_a^2 RM_1 tani.cos(\beta - p_1) - p$$

$$b_1 = \lambda_b^2 RM_{fg} - \frac{1}{2}\lambda_b^2 RM_1 tani.cos(\beta - p_1) - p$$

$$r_1 = -RM_0 tani.cosp_0$$

$$r_2 = -RM_0 tani.sinp_0$$

$$r_3 = -\frac{1}{2}RM_1 tani.cos(p_1 + \beta)$$

$$r_4 = -\frac{1}{2}RM_1 tani.sin(p_1 + \beta) \quad (10)$$

As we will see in Sect. all the other coefficients are dependent on each other. The subscript zero refers to the axisymmetric component and one refers to the bisymmetric component. RM_0 and RM_1 are the characteristic rotation measures that characterise the equivalent total field strengths in axisymmetric B_0 and bisymmetric B_1 components, respectively:

$$RM_{0,1} = 0.81 n_e |B_{0,1}| h \quad (11)$$

From equations (10), the following equations can be directly obtained:

$$RM_0 = \frac{(r_1^2 + r_2^2)^{1/2}}{tani} \quad (12)$$

$$RM_1 = 2\frac{(r_3^2 + r_4^2)^{1/2}}{tani} \quad (13)$$

$$p = \frac{a_1\lambda_b^2 - b_1\lambda_a^2}{\lambda_a^2 - \lambda_b^2} \quad (14)$$

From equations (12), (13) and (14) and using the relation given in (5), we can now find RM_{fg}. We will only consider

the case where the regular magnetic field of the galaxy is purely axisymmetric or bisymmetric. We will discuss four cases and give the expression of RM_{fg} for each of them.

i) When RM_0 =0, i.e., a bisymmetric field, we have p_1=p then:

$$R_{fg} = \lambda^{-2}[b_1 + \frac{1}{2}RM_1 . tani . cos(\beta - p_1) - p_1]$$
$$\beta = arctan\frac{r_4}{r_3} - p_1 \quad (15)$$

Here, p1 doesn't yield the correct mean value; in fact it depends on θ.

ii) When RM1=0 we have p0=p then:

$$RM_{fg} = \frac{a_1 - b_1}{\lambda_a^2 - \lambda_b^2}$$
$$p_0 = arctan\frac{r_2}{r_1} \quad (16)$$

We have four equations for the three physical parameters RM_{fg}, RM_0, and p_0, in the case of two-frequency observations and a purely axisymmetric field. This enables the development of a further criterion for the model's consistency; only values of p_0 belonging to the permissible ranges for p_0 given by equations (14) and (16) are valid. If p_0 truly depends on θ, Eq. (14) provides a reliable estimate of the mean value of p_0.

iii) If the galaxies are strongly inclined, i.e., i=90^0, and take $\psi_0(\theta)$=$\frac{1}{2}\pi$. Then equations (12) and (13) hold. Consider RM_0 =0, then:

$$[RM_{fg} - (r_3^2 + r_4^2)cos(\beta - p_1)]\lambda_b^2 = b_1 + \frac{1}{2}\pi,$$
$$p_1 + \beta = arctan\frac{r_4}{r_3}, \text{ and}$$
$$\frac{a_1\lambda_b^2 - b_1\lambda_a^2}{\lambda_a^2 - \lambda_b^2} \quad (17)$$

iv) If the galaxies are strongly inclined, i.e., i=90^0, and take $\psi_0(\theta)$=$\frac{1}{2}\pi$. Then equations (12) and (13) hold. Consider RM_1 =0, then:

$$RM_{fg} = \lambda_a^{-2}(a_1 + \frac{1}{2}\pi),$$
$$RM_{fg} = \lambda_b^{-2}(b_1 + \frac{1}{2}\pi), \text{ and}$$
$$p_0 = arctan\frac{r_2}{r_1} \quad (18)$$

2.3 Statistical analysis

Substituting equation (3) in (5) and subtracting (5b) from (5a) we get:

$$\psi_a(\theta) - \psi_b(\theta) = (r_0 + r_1cos\theta + r_2sin\theta + r_3cos2\theta + r_4sin2\theta + \ldots)(\lambda_a^2 - \lambda_b^2) + R_{fg}(\lambda_a^2 - \lambda_b^2)$$
$$= (a_0 - b_0) + (a_1 - b_1)cos\theta + (a_2 - b_2)sin\theta + (a_3 - b_3)cos2\theta + (a_4 - b_4)sin2\theta + \ldots \quad (19)$$

Now comparing both sides of equation (19) we get:

$$\frac{a_2}{b_2} = \frac{a_3}{b_3} = \frac{a_4}{b_4} = \frac{a_5}{b_5} = \ldots = \frac{\lambda_a^2}{\lambda_b^2},$$
$$a_1 = (r_0 + RM_{fg})\lambda_a^2 + \psi_0, \text{ and}$$
$$b_1 = (r_0 + RM_{fg})\lambda_b^2 + \psi_0 \quad (20)$$

The independent parameters that have to be found out of the observational data are, $a_1, b_1, r_1, r_2, r_3, r_4, \ldots$.

Now we discuss the uncertainties and errors. The uncertainties is due to the unresolved radial gradient of the regular magnetic field is given by:

$$\sigma_4 = \Delta R\frac{\delta\psi}{\delta R} \quad (21)$$

Now the estimation for each wavelength from the relation given by:

$$\frac{\delta\psi}{\delta R} = \frac{1}{R_2' - R_1'}[(\psi_i^{(2)} - \psi_i^{(1)})^2]^{1/2} \quad (22)$$

where the dash denotes averaging over all azimuthal sectors, R'_1 = 1/2 (R_1 + R_2) and R2 = 1/2(R_2 + R_3) are the median radii of the rings $R_1 < R < R_2$ and $R_2 < R < R_3$ and $\psi_i^{(2)}$ and $\psi_i^{(1)}$ are the measured position angles in the outer and inner ring, respectively.

The error associated with the unresolved azimuthal gradient of the longitudinal magnetic field given by:

$$\psi_3 = \frac{1}{\sqrt{2}}(m_{max} + 1)\psi_{max}\Delta\theta \quad (23)$$

Where ψ_{max} is the amplitude of variations of ψ_i with θ and $\Delta\theta$ the sector opening angle is given by:

$$\psi_{max} = [(\psi_i - \psi_i')^2]^{1/2} \quad (24)$$

for each wavelength and each ring.

In order to determine the quality or reliability of a fit for the Fourier series given in equations (3) and (4) for the observational data, we look at the dimensionless sum given by:

$$S = S_a + S_b$$
$$= \Sigma_{i=1}^{N_a}(\frac{\psi_{ai} - \psi_a(\theta_i)}{\sigma_{ai}})^2 + \Sigma_{i=1}^{N_b}(\frac{\psi_{bi} - \psi_b(\theta_i)}{\sigma_{bi}})^2 \quad (25)$$

which is called the residual, with S_a and S_b being the contributions for the individual wavelengths. Here ψ_{ai} ψ_{bi} are the measured position angles at the azimuth $\theta = \theta_i$ and wavelengths λ_a and λ_b respectively, σ_{ai} and σ_{bi} are the standard deviations of these measurements. N_a and N_b are the respective numbers of the measured points and the functions $\psi_a(\theta)$ and $\psi_b(\theta)$. The values of the coefficients a_n and b_n, which correspond to the minimal possible value of the residual, S_{min}, are the most probable. The model with a given number of coefficients a_n and b_n is said to be reliable with respect to the χ^2 **test** at the confidence level (probability) P if the following inequality holds:

$$S_{min} < OR = \psi_{N-k}^2(P) \quad (26)$$

where N = N_a + N_b is the total number of measurements at both wavelengths and k is the number of independent parameters in the chosen model; P=0.95 corresponds to the 2 σ deviation of a Gaussian random variable.[1]

The azimuthal variations of $\psi_0(\theta)$ should lead to unequal contributions to unequal contributions to the total residual S coming from individual wavelengths, S_a and S_b, with $S_a >> S_b$. The **Fisher test** gives the statistical reliability of this difference, i.e., the difference between S_a and S_b can be considered statistically significant at confidence level P when:

$$\frac{max(s_a, s_b)}{min(s_a, s_b)} > F(N_a - k, N_b - k, P) \quad (27)$$

Where $s_{a,b}$=$S_{a,b}$/($N_{a,b}$-k) and F(n_1,n_2,P) is the Fisher criterion.[4]

Uncertainties of the final parameter estimates are evaluated based on the matrix of the second derivatives:

$$A_{ij} = \frac{\delta^2 S}{\delta x_i \delta x_j}|_{S=S_{min}} \quad (28)$$

where x1=a1, x2=b1 and xn=rn-2 for n>=3. The uncertainty of an estimated parameter xn is given by $\Delta x_n = [(\chi^2 - S_{min})/A_{nn}]^{1/2}$. Choosing the value of χ^2 as corresponding to P=0.95, we obtain the 2σ errors of the final parameters, which are given below.[1]

Off-diagonal elements of the matrix Air determine the covariance matrix:

$$\alpha_{ij} = \frac{A_{ij}}{(A_{ii}A_{jj})^{1/2}}|_{S=S_{min}}, \quad (29)$$

whose elements characterise the reasonability of the choice of the independent parameters of the model: the chosen set of parameters can be considered good when α_{ij}<<1 for i!=j ; $|\alpha_{ij}| = 1$ implies that x_i and x_j cannot be considered as independent parameters.[1]

3. Review of A. Fletcher et al. paper

3.1 The model

The polarisation angle of synchrotron emission is given by:

$$\psi_a(B) = RM(B).\lambda_a^2 + RM_{fg}.\lambda_a^2 + \psi_0(B) \quad (30)$$

where $\psi_0(\theta)$ is the intrinsic polarisation angle, RM_i is the Faraday rotation measure in the galaxy, RM_{fg} is the Faraday rotation in the Milky Way and λ is the wavelength. The cylindrical components of $\boldsymbol{B}$ = (B_r, B_θ, B_z) are expanded in a Fourier series in the azimuthal angle θ,

$$B_r = B_0 sinp_0 + B_1 sinp_1 cos(\theta - \beta_1) + B_2 sinp_2 cos2(\theta - \beta_2)$$
$$B_\theta = B_0 cosp_0 + B_1 cosp_1 cos(\theta - \beta_1) + B_2 cosp_2 cos2(\theta - \beta_2)$$
$$B_z = B_{z0} + B_{z1} cos(\theta - \beta_{z1}) + B_{z2} cos2(\theta - \beta_{2z}) \quad (31)$$

where β_m and β_{zm} are the azimuths where the non-axisymmetric modes are greatest, B_m and B_{zm} are the amplitudes of the mode with azimuthal wave number m in the horizontal and vertical fields, p_m is the pitch angle of the mth horizontal Fourier mode. The expression for $\psi_0(\theta)$ is given by equation (7). RM is related to $B_{||}$ and involves other parameters such as the thermal disc height and $\xi\lambda$, which will lead us to the discussion of depolarisation, hence, is neglected.[3]

Here, it is important to notice that RM depends on magnetic field components along the line of sight, whereas θ depends on those in the sky plane. We can therefore derive all three components of B by fitting equation(30). The amplitudes of the Fourier modes are determined by fitting in terms of the variables R_m since the observed polarisation angle depends on RM, or the product of the magnetic field strength, thermal electron density, and the path length, whose dimension is given by rad m^{-2}:

$$R_m = 0.81(\frac{B_m}{1\mu G})(\frac{< n_e >}{1cm^{-3}})(\frac{L}{1pc}) \quad (32)$$

Faraday rotation is scaled by a wavelength-dependent factor $\xi\lambda \leq 1$, so

$$RM_i = \xi\lambda RM_d, \quad (33)$$

where RM_d is the Faraday rotation measure produced through the whole disk thickness, and $\xi\lambda$ can be understood as the fraction of the disk thickness transparent to polarised emission at a wavelength λ. In Sect. 6, we discuss depolarising mechanisms in detail, and from models of the observed depolarisation, we adopt ξ_6 = 1.0, ξ_{11} = 0.96 and ξ_{20} = 0.75. The fitted parameters describing the magnetic field were found to be rather insensitive to the adopted values of $\xi\lambda$.

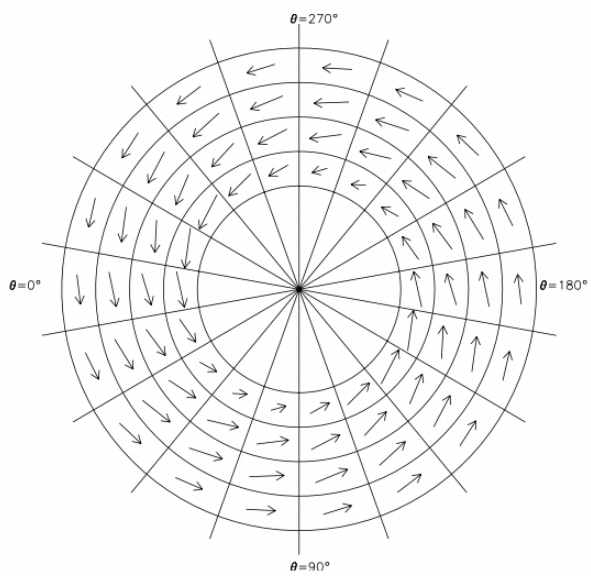


Fig. 3. Face-on view of M 31 showing sectors and regular magnetic field vectors obtained from the fits shown in Table 2. The grid radii are 6, 8, 10, 12 and 14 kpc. The length of the vectors is proportional to $\boldsymbol{B}$.[2]

The reliability of the fit is determined by the χ^2 test and the Fischer criterion which has already been discussed in Sect. 2.3.

4. Conclusion

Both papers have been reviewed, and there is a subtle difference between the two papers that has been noted.

In D. Sokoloff et al. paper, we classified the model into two techniques. Wherein, in the first one, we tried to fit the data for RM and the second for position angles. Then we combined both of them to obtain the magnetic field parameters (including pitch over different radii).

In A. Fletcher et al. paper, the position angle was fitted by taking it as a function of magnetic field. The paper then discussed the method of depolarisation to get the complete information about the magnetic field. Which additional information gave the height of the synchrotron disc, height of the thermal disc, method to determine $\xi\lambda$, <n_e> and information about the interstellar medium.

Hence, A. Fletcher et al. paper has built upon the techniques used in D. Sokoloff et al. paper to achieve more information about the magnetic fields and interstellar medium of spiral galaxies.

5. Acknowledgements

This work was supported by the National Institute of Science Education and Research (NISER), Odisha, India.

I want to thank my supervisor, Dr. Tuhin Ghosh, for his active guidance and assistance at every stage when it was required. He helped me understand some fundamental concepts through his collaboration, encouragement, and inspiration throughout the project.

Finally, I thank my family and friends for their encouragement, support and keeping me enthusiastic throughout the project.

References

[1] D. Sokoloff, A. Shukurov, and M. Krause, Astron. Astrophys. 264, 396-405 (1992)
[2] A. Fletcher, E.M. Berkhuijsen, R. Beck, and A. Shukurov, A&A 414, 53-67 (2004), DOI:10.1051/0004-6361:20034133
[3] E.M. Berkhuijsen, C. Horellou, M. Krause, N. Neininger, A.D. Poezd, A. Shukurov and D.D. Sokoloff, Astron. Astrophys. 318, 700-720 (1997)
[4] G.A. Korn and T.M. Korn, Mathematical Handbook for Scientists and Engineers (Dover Publications, Inc. Mineola, New York)

FINAL PROJECT REPORT

TRANSITING EXOPLANETS USING TESS DATA

H SHISHIR
BATCH 21’
NISER BHUBANESHWAR

INDEX

SL. NO.	CONTENT	PAGE NO.
1.	ABSTRACT	3
2.	INTRODUCTION	3
3.	TESS	3
4.	TESS MISSION	4
	4.1 PRIMARY MISSION	
	4.2 FIRST EXTENDED MISSION	
	4.3 SECOND EXTENDED MISSION	
	4.4 KEPPLER vs. TESS	
5.	DATA PIPELINE	6
6.	CREATING LIGHTCURVE USING LIGHT CURVE FILES	7
7.	MAKING CUSTOMIZED APERTURES TO ANALYZE LIGHT CURVES USING TARGET PIXEL FILES (TPFs)	9
8.	NOISE REMOVAL	11
9.	CROWD CORRECTION	14
10.	CONCLUSION	16
11.	ACKNOWLEDGEMENT	17
12.	REFERENCES	17

TRANSITING EXOPLANETS USING TESS DATA

1. ABSTRACT

We focus on understanding TESS data on transiting exoplanets. Understanding how TESS is different from its counterparts. The pipeline describes the flow of data and how it is processed using the data pipeline. `matplotlib` is used to achieve embedded plots which is an extension to `numpy`. In order to analyze this data we use the `lightkurve` package. It includes all the functions on accessing Light Curve Files (LCFs) and Target Pixel File (TPFs) which gives data on flux which can be extrapolated. We can also interact with the fies to analyze these files and can perform operations for reducing the background noise and crowding.

2. INTRODUCTION

Observing exoplanets helps us understand how different planetary systems are formed and allows us to determine whether or not we understand these processes. There are various methods to detect exoplanets. One of them being the transit method. The transit method is a photometric method that aims to indirectly detect the presence of one or more exoplanets in orbit around a star.

The Transiting Exoplanet Survey Satellite (TESS) is a NASA-sponsored Astrophysics Explorer-class mission that is performing a near all-sky survey to search for planets transiting nearby stars. The primary goal of TESS is to discover planets smaller than Neptune that transit stars bright enough to enable follow-up spectroscopic observations that can provide planet masses and atmospheric compositions.

3. TESS

TESS observes from a unique elliptical high Earth orbit (HEO) that provides an unobstructed view of its field to obtain continuous light curves and a more stable platform for precise photometry than the low Earth orbit. The launch carried the observatory to a parking orbit inclined by 28.5 degrees.[2]

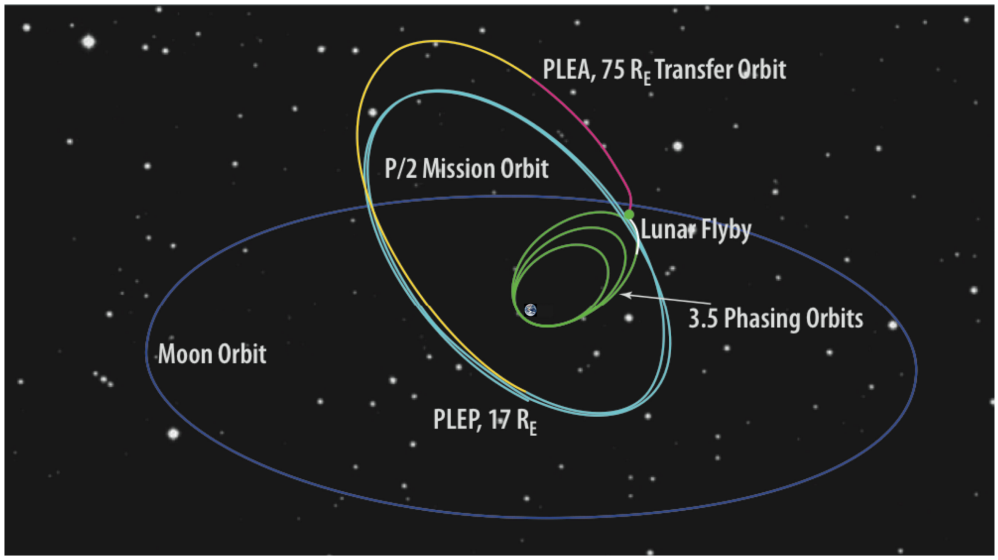


FIG 1: *Schematic of maneuvers and encounters leading to the final TESS orbit (light blue). The observatory orbits with a period of ~13.7 days in a 2:1 resonance with the Moon. PLEA and PLEP are the post-lunar-encounter-apogee and -perigee, respectively. Image Credit: Ricker et al. (2015)*

4. TESS MISSION

4.1. PRIMARY MISSION

TESS had begun regular science operations on July 25, 2018. In the 2-year prime mission, TESS monitored over 200,000 main-sequence dwarf stars with four wide-field optical CCD cameras to detect periodic drops in brightness caused by planetary transits. Photometry of these pre-selected targets was recorded every 2 minutes. TESS also obtained full-frame images (FFIs) of the entire, four camera field-of-view, which had 24 x 96 degrees, at a cadence of 30 minutes to facilitate additional science.[2]

Specifically, the primary science requirements of the TESS mission were as follows:

1. Search over 200,000 stars to discover planets with periods <10 days and radii <2.5 Earth radii orbiting the brightest stars in the solar neighborhood and discover planets with radii at least 2.5 Earth radii distributed across the celestial sphere.
2. Search for transiting planets with periods up to 120 days among the ~10,000 stars in regions surrounding the ecliptic poles.
3. Determine the masses for at least 50 planets with radii <4 Earth radii.

4.2. FIRST EXTENDED MISSION

In July of 2020 TESS entered into its first extended mission that started. In this extension the TESS mission changed to become a more community focused science mission. A new 20-second cadence data mode was introduced, and the FFI cadence changed from 30-minutes to 10-minutes.

4.3. SECOND EXTENDED MISSION

TESS will enter into its second extension of the mission in September, 2022. During this second extension the FFI cadence will change to 200 seconds.

4.4. KEPLER vs. TESS

NASA's Kepler Mission was, during its 4 year prime mission from 2009-2013, a statistical transit survey designed to determine the frequency of Earth-sized planets around other stars. Kepler revealed thousands of exoplanets orbiting stars in its 115 square degree field-of view, which covered about 0.25 percent of the sky. While Kepler was revolutionary in its finding that Earth-to-Neptune-sized planets are common, the bulk of the stars in the Kepler field lie at distances of hundreds to thousands of parsecs, making it difficult to obtain ground-based follow-up observations for many systems.[2]

The TESS Mission is designed to survey over 85% of the sky (an area of sky 400 times larger than covered by Kepler) to search for planets around nearby stars (within ~200 light years). TESS stars are typically 30-100 times brighter than those surveyed by the Kepler satellite. Planets detected around these stars are therefore far easier to characterize with follow-up observations, resulting in refined measurements of planet masses, sizes, densities, and atmospheric properties.[2]

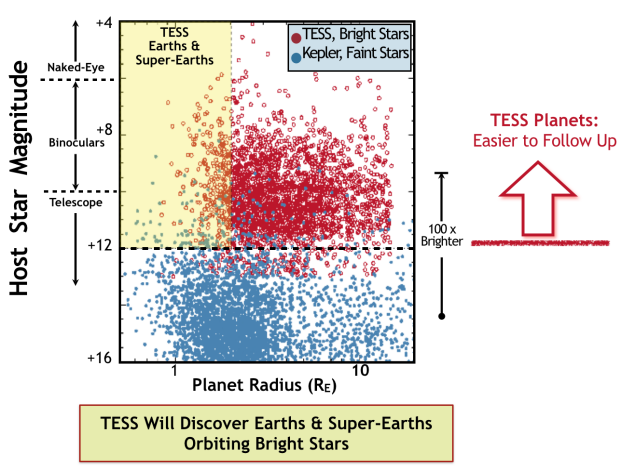


FIG. 2: *Stars detected by Kepler and TESS*

Image Credits: MIT/TESS/NASA

5. DATA PIPELINE

The TESS Mission is led by the Massachusetts Institute of Technology (MIT), which is responsible for the overall direction of the mission. The TESS Science Office (TSO) is a partnership between MIT's Physics Department and Kavli Institute for Astrophysics and Space Research and the Smithsonian Astrophysical Observatory. The TESS Science Office analyzes the science data and organizes the co-investigators, collaborators, and working groups. NASA's Goddard Space Flight Center (GSFC) provides project management, systems engineering, and safety and mission assurance. The TESS Science Support Center (TSSC) at GSFC operates the Guest Investigator Program. The spacecraft is operated by Northrop Grumman Innovation Systems. This includes mission operations at the Mission Operation Center. The data are processed by the Science Processing Operations Center (SPOC) at NASA Ames Research Center. All TESS raw and processed data are made publicly available through the Mikulski Archive for Space Telescopes (MAST), operated by the Space Telescope Science Institute (STScI).[2]

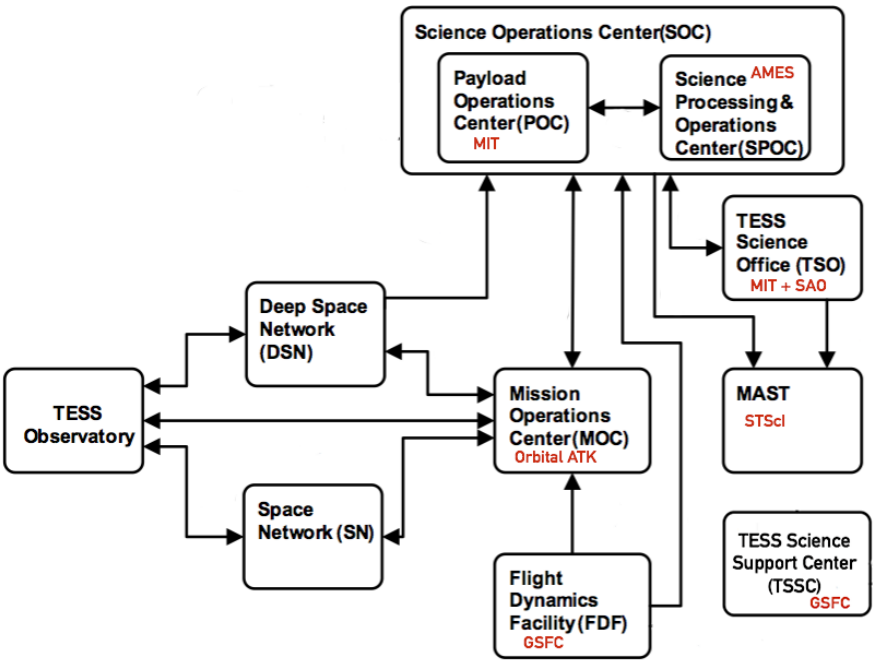


FIG. 3: *TESS Data Pipeline*

Image Credits: MIT/TESS/NASA

6. CREATING LIGHTCURVE USING LIGHT CURVE FILES

The following program plots the light curve for the system HATS-71b.

```
!pip install lightkurve

import lightkurve as lk
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

search_ffi = lk.search_tesscut('HATS-71b')
search_tpf = lk.search_targetpixelfile('HATS-71b')
search_lcf = lk.search_lightcurve('HATS-71b')

search_lcf

search_lcf_refined = lk.search_lightcurve('HATS-71b', author="SPOC", exptime=120)
search_lcf_refined

lcf = search_lcf_refined.download_all()
lcf[0]

pdcsap = lcf[0].pdcsap_flux
sapflux = lcf[0].sap_flux
ax = lcf[0].plot(column='sap_flux', normalize=True, label="SAP");
lcf[0].plot(ax=ax, column='pdcsap_flux', normalize=True, label="PDCSAP");

ax = lcf[0].plot()
ax.set_title("PDCSAP light curve of  HATS-71b")

lcf[0].remove_nans().flatten(window_length=501).fold(period=3.7955, epoch_time=1325.6501).bin(time_bin_size=0.025).plot();
```

Here we first access the lightcurve files of HATS-71b. In order to obtain the light curve for one of the search results we need to download the data of that particular file. The parameters to access the file are:

1. Name of the system for which we want to plot the light curve.
2. Name of the author.
3. The exposure time. Exposure time refers to the length of time the camera collects light from your sample.

Once we download the files we plot SAP Flux and PDCSAP Flux. These are plotted together so that they can be compared.

- **Simple Aperture Photometry (SAP) flux:** The SAP light curve is calculated by summing together the brightness of pixels that fall within an aperture set by the TESS mission. This is often referred to as the optimal aperture, but in spite of its name, it can sometimes be improved upon! Because the SAP light curve is a sum of the brightness in chosen pixels, it is still subject to systematic artifacts of the mission.
- **Pre-search Data Conditioning SAP flux (PDCSAP) flux:** SAP flux from which long term trends have been removed using so-called Co-trending Basis Vectors (CBVs).

PDCSAP flux is usually cleaner data than the SAP flux and will have fewer systematic trends.

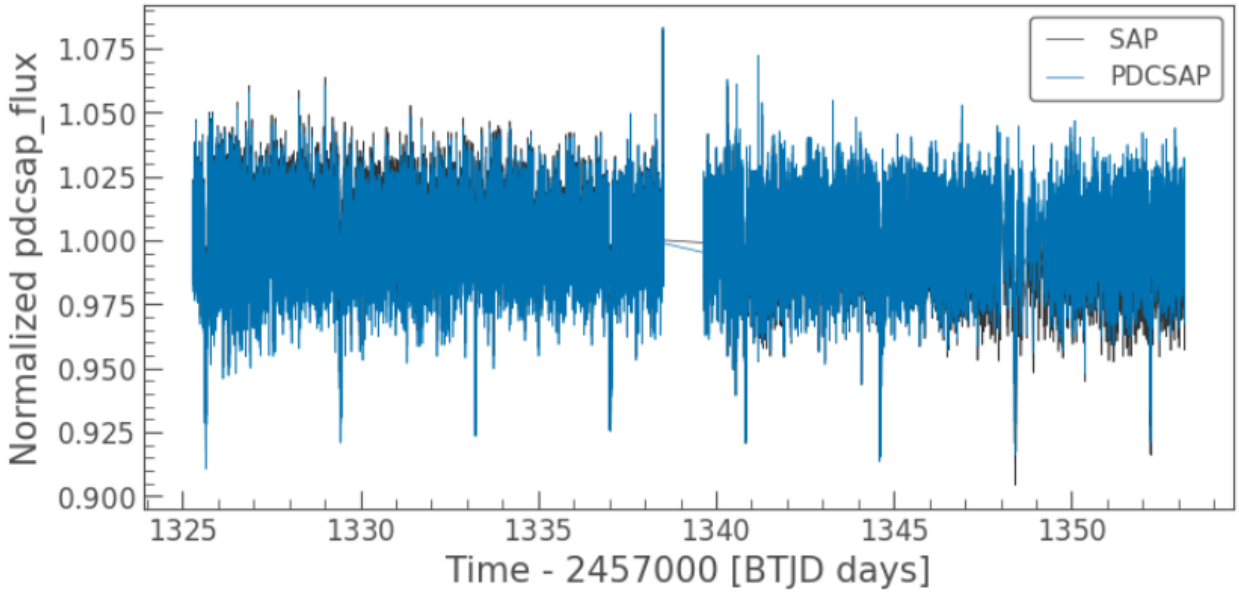


FIG. 4: *Plot of SAP and PDC SAP flux of HATS-71b*

We can perform operations on the above raw data in order to obtain the transit. Few of the operations are:

- **flatten():** Remove long term trends using a Savitzky–Golay filter
- **remove_outliers():** Remove outliers using simple sigma clipping
- **remove_nans():** Remove infinite or NaN values (these can occur during thruster firings)
- **fold():** Fold the data at a particular period
- **bin():** Reduce the time resolution of the array, taking the average value in each bin.

The plot below represents the transit after performing the above operations.

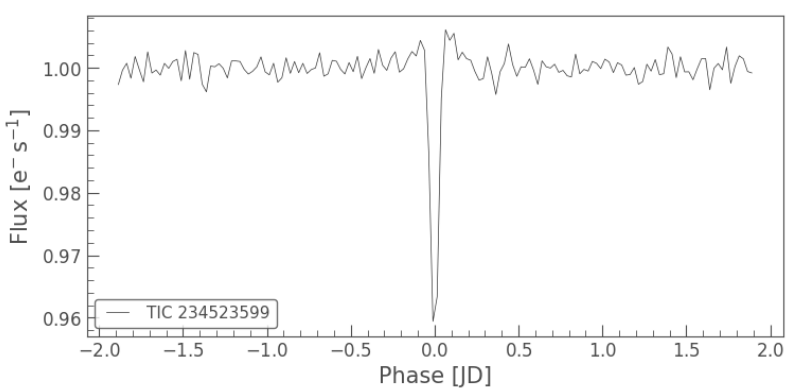


FIG. 5: *Transit of HATS-71b*

7. MAKING CUSTOMIZED APERTURES TO ANALYZE LIGHT CURVES USING TARGET PIXEL FILES (TPFs)

The program below shows how TPFs are analyzed and masked in order to obtain light curves.

```
!pip install lightkurve

%matplotlib inline
import lightkurve as lk
import matplotlib.pyplot as plt
import numpy as np

search_result = lk.search_targetpixelfile('TIC 259377017')
search_result[0]

tpf_file = search_result[0].download(quality_bitmask='default')
tpf_file

tpf_file = lk.search_targetpixelfile('TIC 259377017', mission="SPOC", sector=3).download(quality_bitmask='default')
tpfs=search_result.download(cutout_size=10)

tpf_file.plot();
tpf_file.plot(aperture_mask=tpf_file.pipeline_mask);
tpf_file.pipeline_mask

target_mask = tpf_file.create_threshold_mask(threshold=20, reference_pixel='center')
n_target_pixels = target_mask.sum()
n_target_pixels
tpf_file.plot(aperture_mask=target_mask, mask_color='r');

aper_new = np.zeros(tpfs.shape[1:], dtype=bool)
aper_new[3:9, 4:9] = True
tpfs.plot(aperture_mask=aper_new, mask_color='red')
target_lc_new = tpf_file.to_lightcurve(aperture_mask=aper_new)
target_lc_new

tpf_file.plot();
target_lc_new.plot();
```

These TPFs are downloaded for a particular system. In order to obtain the light curve for one of the search results we need to download the data of that particular file. The parameters to access the file are:

1. TIC number of the planet.
2. Name of the author.
3. The sector being observed by TESS.

We can even specify the cutout size to build an effective mask. Masking refers to the process of choosing particular target pixel files. This helps to select the particular object of interest and eliminate the background noise and correct crowding.

The below image shows the pixels given by the author.

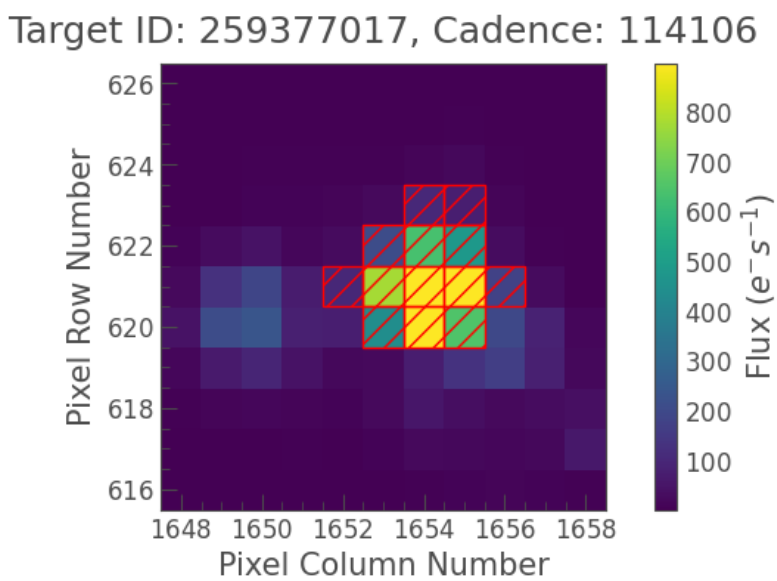


FIG. 6: *TPF Image of TIC 259377017*

If we let the pipeline select the pixels then it will select the brightest or the region of pixels which have very high flux. This can sometimes cause a problem as it might not be our object of interest and might include a lot of background noise.

The image below shows the pixels in red which have been masked, i.e., we have created our own photometry aperture to plot the light cure.

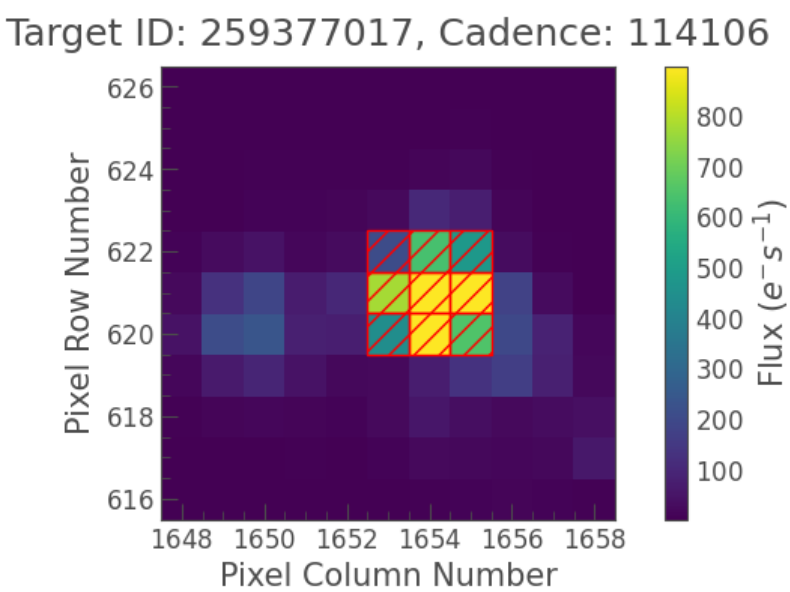


FIG. 7: *Masked TPF Image of TIC 259377017*

The light curve for the custom aperture is given by:

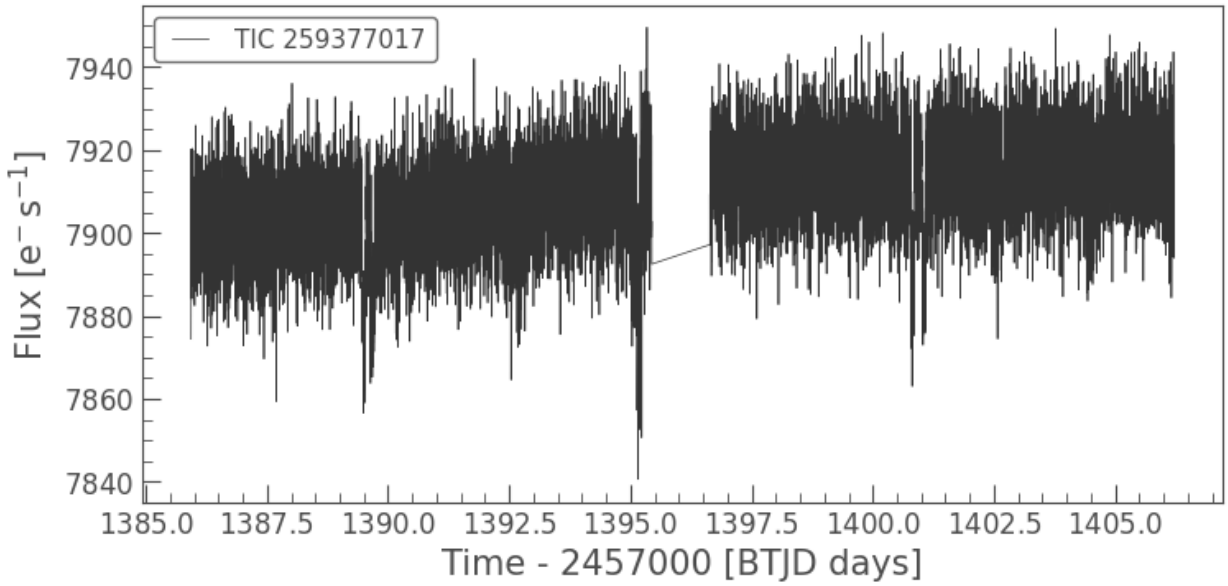


FIG. 8: *Light curve for masked TPF Image of TIC 259377017*

8. NOISE REMOVAL

We use correctors to remove scattered light and noise from our data. We will focus on CBVCorrector. *(The program for Noise Removal has been attached in the mail)*

- **CBVCorrector:** Co-trending Basis Vectors (CBVs) are generated from the most common systematic trends observed in each Sector. Each TESS CCD and Camera has its own set of CBVs. There are three basic types of CBVs,
    a. Single-Scale: Contains all systematic trends combined in a single set of basis vectors. Generally speaking, a single-scale CBV performs better at preserving longer period signals.
    b. Multi-Scale: Contains systematic trends in specific wavelet-based band passes. There are usually three sets of multi-scale basis vectors in three bands. This type of correction performs better when the periods are close to the transiting planet durations, and help to preserve the signal.
    c. Spike: Contains only short impulsive spike systematics. This correction is applied to remove short impulsive systematic signals.

Other correctors are RegressionCorrector and PLDCorrector.

- **RegressionCorrector:** Here the light curve is de-trended against vectors that we think are predictive of the systematic noise. For FFI data, we have to select/define an aperture for our object of interest. Pixels outside of this aperture are used to create vectors that are predictive of the noise. These vectors are then removed from the data.
- **PLDCorrector:** Pixel Level De-correlation (PLD) works by identifying a set of trends in the pixels surrounding the target star, and performing linear regression to create a combination of these trends that effectively models the systematic noise introduced by spacecraft motion/scattered light. This noise model is then subtracted from the uncorrected light curve. It is similar to the RegressionCorrector.

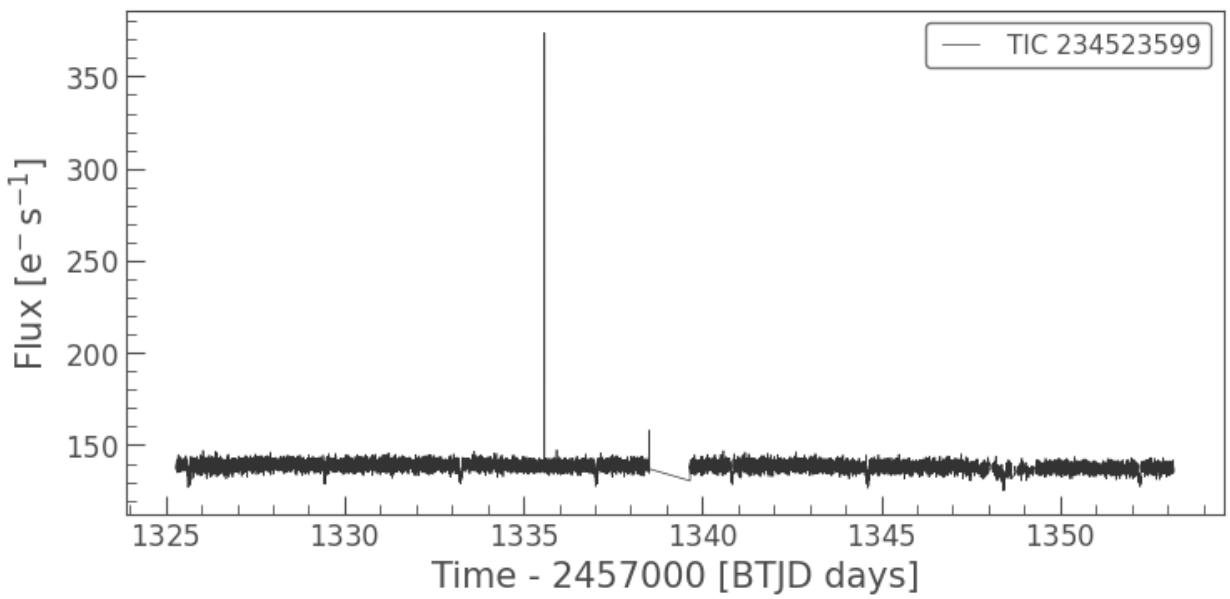


FIG. 9: *Light curve of TIC 234523599*

We have plotted the light curve for TIC 231523599.

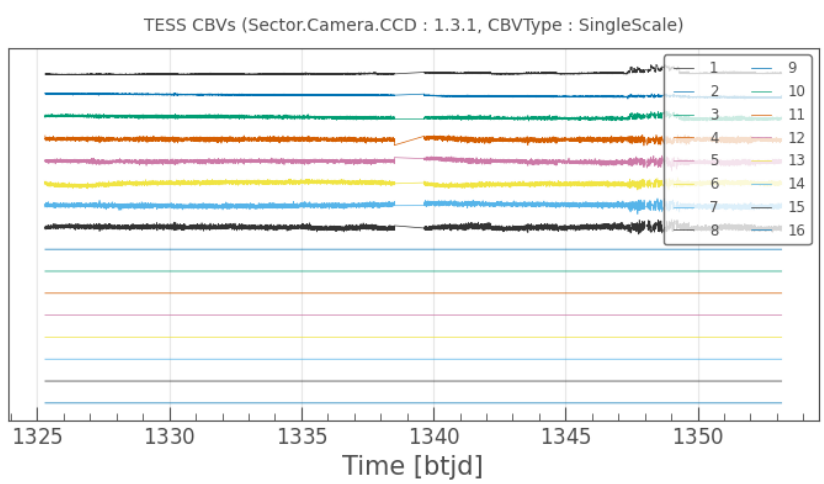


FIG. 10: *TESS CBVs for TIC 234523599*

The first several CBVs contain most of the systematics. The latter CBVs pose a greater risk of injecting more noise than helping. The default behavior in CBVCorrector is to use the first 8 CBVs. Next, we need to decide what kind of CBV we want to apply for. Since we want to try not to overfit, we are going to use the Multi-Scale and Spike options.

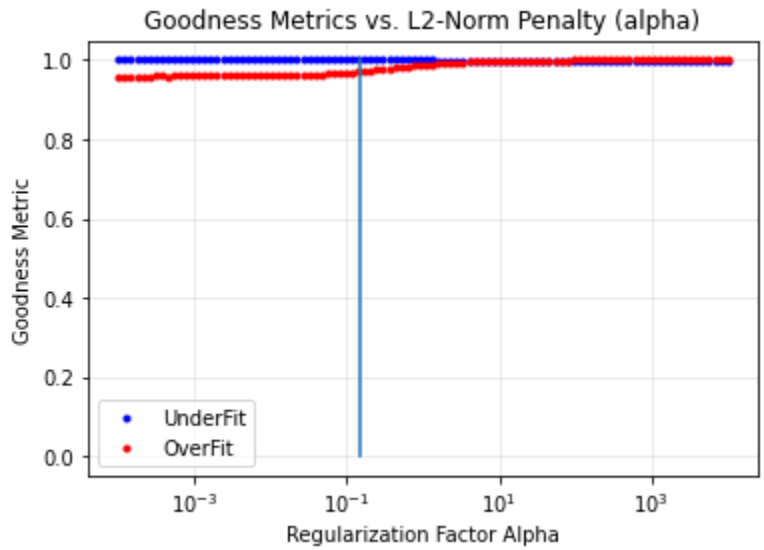


FIG. 11: *Goodness Metrics of TIC 234523599 Data*

Now we want to check that the application of these vectors will not under or over-fit our data. There is a handy way of checking this via the use of a goodness_metric_scan_plot. A "Goodness Metric" value of 0.8 and above tends to indicate a good fit to the data.The above plot indicates that a regularization factor alpha value of 1.5e-1 is appropriate for our data.

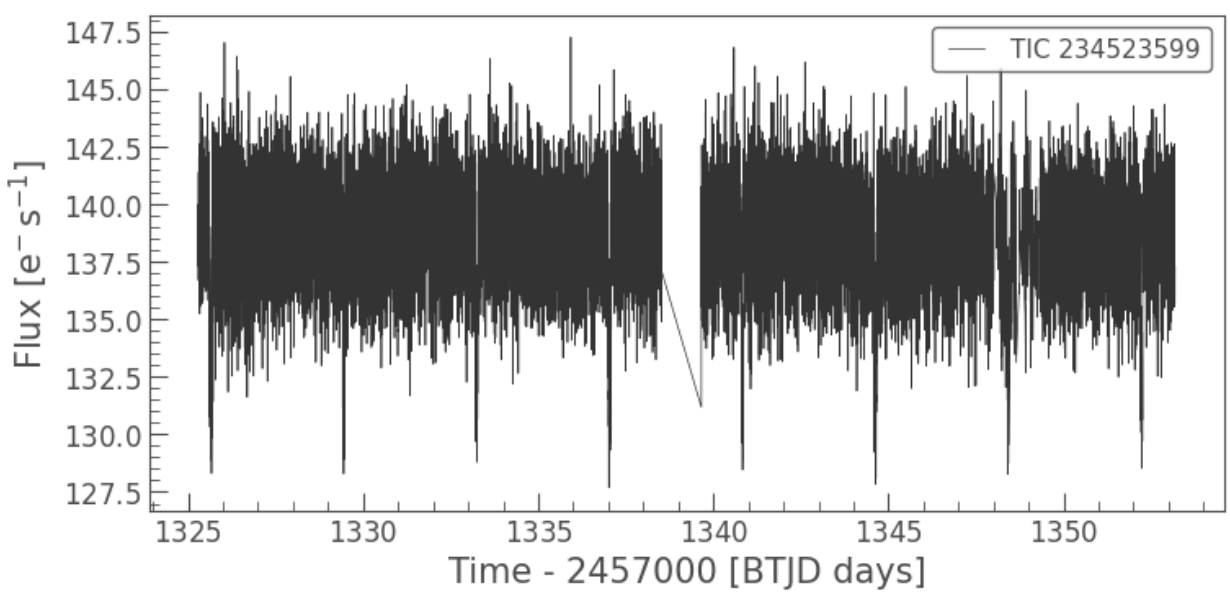


FIG. 12: *Light curve of TIC 234523599 after applying CBVCorrection*

We can see that a correction was applied which has removed some of the slope from our data and also labeled the big spike as an outlier.

9. CROWD CORRECTION

For exoplanets, if the excess flux is not removed, it can cause a decrease in the apparent planet transit depth and lead to a systematic underestimation of the planet radii. LightCurve Objects have been corrected for this crowding via the data processing pipeline developed by the Science Processing Operations Center (SPOC). The correction however is applied only to the PDCSAP flux and not the SAP flux. *(The program for Noise Removal has been attached in the mail)*

The crowding correction applied focuses on two parameters:

- The crowding metric: This reflects what fraction of the flux in the aperture is due to the target itself, not the nearby light sources.
- The flux fraction: Similar to excess flux leaking into the aperture, a fraction of the PRF of the target may not be captured in it. To account for this missing fraction, the flux fraction is computed.

We plot SAP and PDCSAP of a binary star system WR21a:

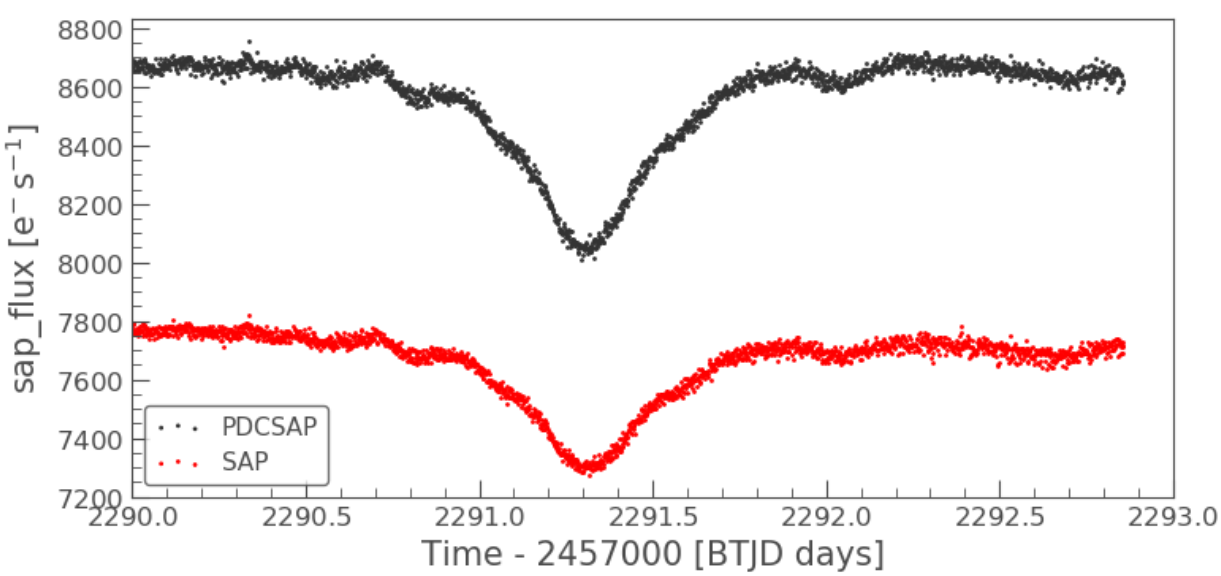


FIG. 13: *Plot SAP and PDC SAP flux of WR21a*

The above plot clearly indicates the significant difference in amplitude between the PDCSAP and SAP flux. The plot illustrates how much of a difference the flux fraction and crowding metric can change the absolute magnitudes of the light curves. Now we plot the normalized PDC SAP and SAP flux to show that the transit depth has changed.

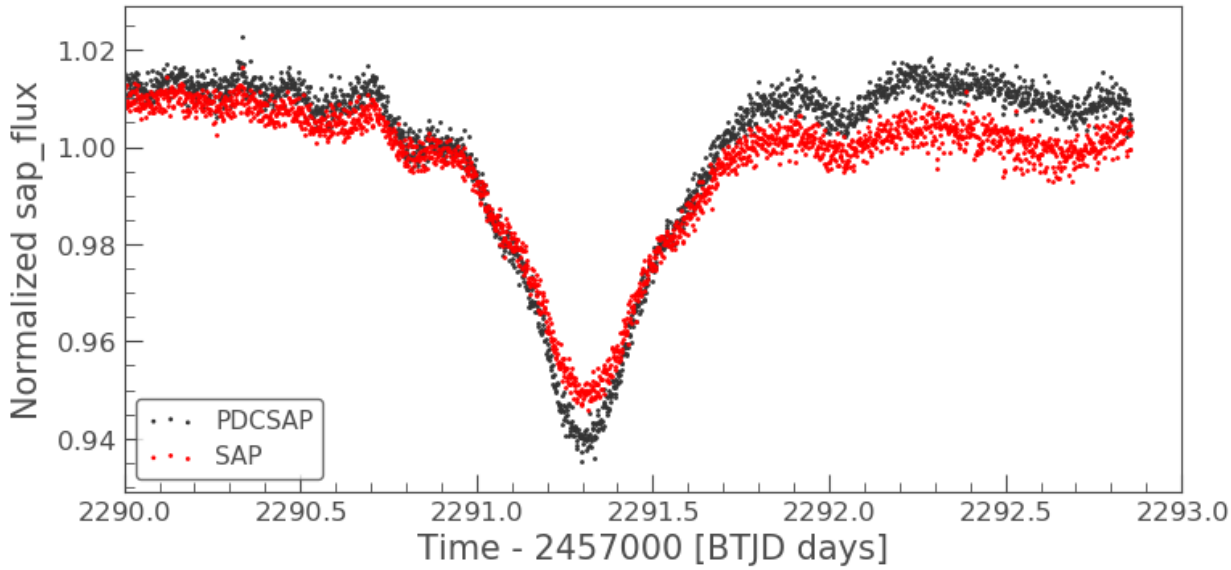


FIG. 14: *Plot of normalized SAP and PDC SAP flux of WR21a*

Now we apply crowd PLDCorrections and compare each one of them.

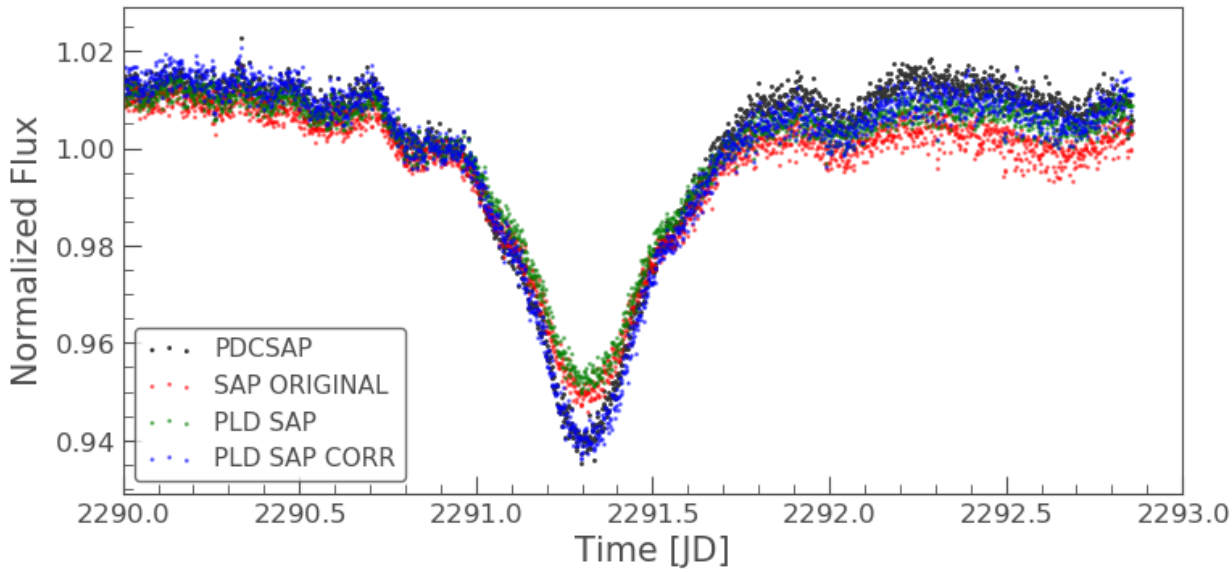


FIG. 15: *Comparison of SAP ORIGINAL, PDC SAP, PLD SAP CORRECTION of WR21a*

It can clearly be seen that PDCSAP and PLD SAP CORR are very closely related. Even after normalization a significant difference can be seen in the transit depth of SAP ORIGINAL and PDCSAP.

Now we can even apply CBVCorrection and observe the difference.

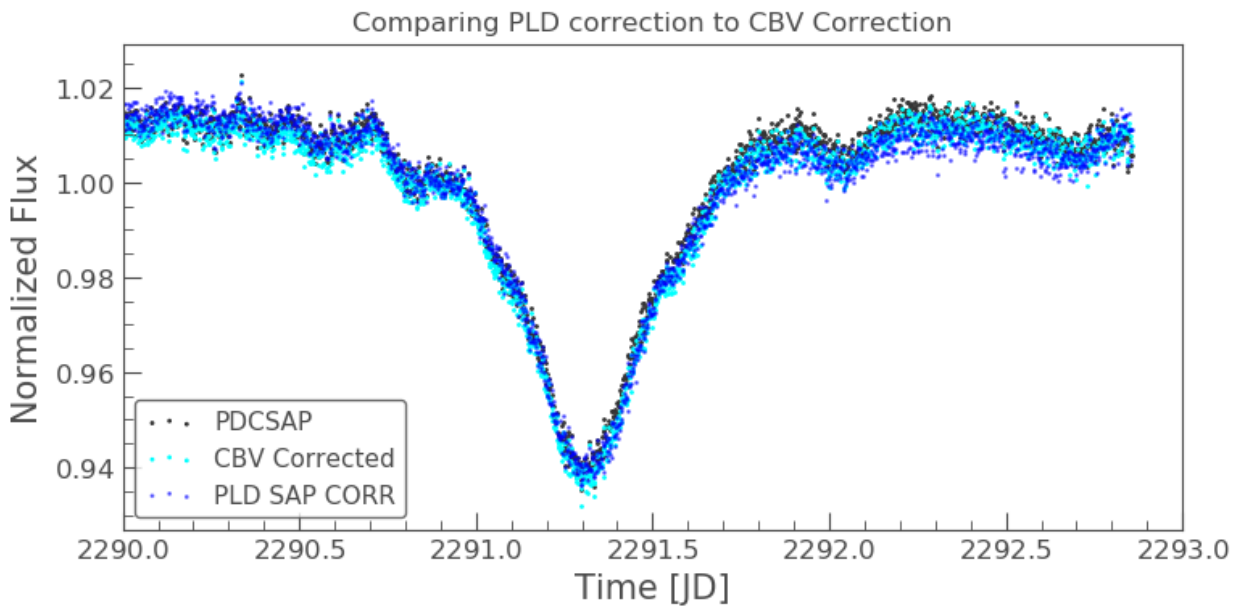


FIG. 16: *Comparison of PDC SAP, CBV CORRECTION and PLD SAP CORRECTION of WR21a*

We can notice that there is a significant overlap of the curves in the transit region and can even notice how closely PDC SAP, PLD Corrected SAP and CBV Corrected SAP are related. Theoretically they hold their own differences due to the methods used to achieve each of them.

10. CONCLUSION

We have seen how exoplanets are detected using the transit method, especially using TESS Data. Learned to analyze the light curves and perform operations on them. We have seen how SAP Flux can be processed and understood the difference between PDC SAP, PLD Corrected SAP and CBV Corrected SAP. Accessing LCFs and TPFs and analyzing data from them. Realized the importance of crowd correction.

11. ACKNOWLEDGEMENT

I would like to thank my peers and colleagues for encouraging and keeping me enthusiastic throughout the project. I wish to show my appreciation to Mr. Rahul Arora who has helped me throughout the project, who has given me insights on programming and troubleshooting and also analyze conceptual details. I extend my special thanks to Prof. Jayesh Goyal who has given me the opportunity to work on this project and be a part of his excellent team. This project has given me an opportunity to explore the vast domains of exoplanet research.

12. REFERENCES

1. C. A. Haswell (2010), "Transiting Exoplanets", *Publications of Cambridge*.
2. https://heasarc.gsfc.nasa.gov/docs/tess.html (Accessed on 02/08/2022)
3. https://mast.stsci.edu/portal/Mashup/Clients/Mast/Portal.html (Accessed on 02/08/2022)
4. https://tess.mit.edu/publications/#1553264481664-5d01f22e-c922 (Accessed on 02/08/2022)

Course Work

Format: Course — Credits — Topics Covered — Book(s) Used — Grade (out of 10)

- **Physics Theory Courses:**

-> P201-Classical Mechanics-I — 8 — Principle of virtual work; constraints; D'Alemberts principle, Generalized coordinates, velocities and momenta; Euler-Lagrange formulation, Two-body central force problem (reduced mass); planet orbits; virial theorem, Collisions and scattering, CM and Lab frames, scattering crosssection., Motion in non-inertial frames; Coriolis force, Principle of least action; formulation by Maupertuis, Euler, Hamilton; Liou- villesstheorem, Hamiltons equations; Poisson brackets, Canonical transformations; Hamilton-Jacobi equation; generating functions; Symmetries and conservation laws, Small oscillations; normal modes — Classical Mechanics by H. Goldstein, C. P. Poole, J. Safko, Pearson Education Dorling Kinderslay, 3rd Ed. — 9

-> P202 Mathematical Methods-I — 8 — Vector Calculus, curvilinear coordinates, Linear vector spaces, linear operators in linear vector spaces, Hermitian, projection and Unitary operators, normal matrices and diagonalisation, Fourier series, Fourier and Laplace transforms, 1st & 2nd order differential equations, power series solution, special functions (Hermite, Legendre, Bessel, Laguerre, hypergeometric), Partial differential equations, separation of variables — Mathematical Methods in the physical sciences by M. L. Boas, Mathematical Methods For Physicists by G. B. Arfken and H. J. Weber — 9

-> P204 Electromagnetism-I — 8 — Introduction to electrostatics and Green function formalism, Laplace and Poisson equations, boundary value problems, Dielectrics, Polarization, electric displacement, Steady currents, Lorentz force; magnetostatic (including vector potentials), magnetic materials, Time-varying fields, Faraday's law, displacement current, Maxwell's equations, electromagnetic waves — Classical Electrodynamics by J. D. Jackson (3rd Edition), Introduction to Electrodynamics by D. J. Griffiths (3rd Edition) — 10

-> P205 Mathematical Methods-II — 8 —Functions of a complex variable, analytic functions, residue theorem and applications, conformal mapping, Taylor and Laurent series, analytic continuation, special analytic functions, Gamma functions, method of steepest descent, Hilbert space, Differential operators and Sturm-Liouville theory, Greens functions, Generalized functions, Cartesian tensors, 4-vectors and 4-tensors, Elements of Group Theory — Mathematical Methods for Physicists by G. B. Arfken & H.J. Weber, Complex Variables and Applications (9th Edition) by James Ward Brown, Ruel V Churchill — 8

-> P206 Quantum Mechanics-I — 8 — Schroedinger equation, one-dimensional problems, Central potentials, Hydrogen atom, 3D harmonic oscillator, Hilbert space formalism (introductory level), operator method for simple harmonic oscillator, General treatment of angular momentum, spin, addition of angular momentum, Approximation methods: Time-independent perturbation theory, degenerate perturbation theory; Ritz variational method, WKB — Introduction to Quantum Mechanics by D. J. Griffiths, Principles of Quantum Mechanics by R. Shankar, Modern Quantum Mechanics by J. J. Sakurai — 9

-> P207 Linear Optics — 4 — Plane waves, Spherical waves, relation of wave optics and ray optics, Interference: Single & multiple-beam interference, Fabry-Perot, Mach-Zehnder, Michelson interferometer, Spatial & temporal coherence, Diffraction: Introduction to Fourier transform, Fresnel and Fraunhoffer integral, Fourier analysis and angular spectrum. Examples of single-slit, multiple-slit, circular aperture; Lens as Fourier transforming element, Gaussian Beam propagation, Laguerre-Gaussian and Hermite Gaussian beam propagation — Optics by Ajoy Ghatak (5th Edition), Tata McGraw Hill, Optics by Eugene Hecht (5th Edition), Pearsons — 9

-> P301 Electromagnetism-II — 8 — Review of Maxwell's equations, Poynting's theorem, electromagnetic energy momentum tensor, Wave propagation in conductors and dielectrics, Reflection, Refraction, Polarization, Total internal reflection, Attenuation of waves in metals, Brewster's angle, Lorentz theory of dispersion, waveguides, fibers and plasma, Gauge transformations and gauge invariance, electromagnetic potentials Lienard-Weichert potentials, radiation from an accelerated charge, Larmor formula, bremsstrahlung and synchrotron radiation, Multipole radiation, scattering by free charges, Time permitting introduction to magnetohydrodynamics/covariant formulation —Classical Electrodynamics by J. D. Jackson (3rd Edition), Introduction to Electrodynamics by D. J. Griffiths (3rd Edition) — 9

-> P302 Statistical Mechanics — 8 — Basics of Probability Theory: Probability distribution, cumulants, central limit theorem; laws of large numbers, Fundamentals of statistical mechanics: Phase space and Liouville theorem; microscopic definition of entropy, ergodic hypothesis, Ensembles theory: Microcanonical, canonical and grand canonical ensembles, Gibbs Paradox, Energy and density fluctuations. Application to ideal gases, spin and non-interacting systems, Review of thermodynamics: Laws of thermodynamics and entropy, Thermodynamic potentials and thermodynamic stability, Quantum Statistical Mechanics: Ideal quantum gases; Bose and Fermi distribution; phonons, photons; Fermi sea; density matrix formulation. Examples: electrons in metal, black body radiation, Bose-Einstein condensation and white dwarf — Statistical Physics of Particles by M. Kardar, Statistical Physics of Fields by M. Kardar — 9

-> P303 Quantum Mechanics-II — 8 — Review of Hilbert space formalism for quantum mechanics, Review of time independent perturbation theory, bound state perturbation theory, Time dependence in QM and Time-dependent perturbation theory, Scattering theory, Greens function methods, Path integral in non-relativistic theory, Relativistic wave equations: Klein-Gordon and Dirac Equations, Dirac particle in presence of an electromagnetic field leading to g = 2, Identical particles, Pauli exclusion principle — Modern Quantum Mechanics by J. J. Sakurai, Advanced Quantum Mechanics by J. J. Sakurai, Principles of Quantum Mechanics by R. Shankar—8

-> P304 Special Theory of Relativity — 8 — Galilean relativity, Newtonian mechanics, Electrodynamics and inconsistency with Galilean relativity, ether and experiments for its detection, failure to detect ether. Measurement of velocity of light in moving frames. Lorentz, Poincare and developments towards relativity, Einstein's special theory: Constancy of velocity of light as a postulate. Derivation of Lorentz transformation. Length contraction and time dilation. Mass- energy relation, Doppler shift. Minkowski space-time diagram, boosts as complex rotations in Minkowski space, Four dimensional space-time continuum, Lorentz transformations as coordinate transformations, vectors, scalar product, scalars, tensors, contravariant and covariant objects, laws of physics as tensor equations, Mechanics, hydro-dynamics and electrodynamics as tensor equations, Beyond special relativity: Inertial and gravitational mass, Equivalence principle — Introduction to Special Theory of Relativity by Resnick — 8

-> P306 Introduction to Condensed Matter Physics — 8 — General introduction, Drude and Sommerfeld model, Crystal structure; x-ray diffraction, Cohesive energy, Blochs theorem; Band theory nearly free electrons; tight binding approximation; semi-classical dynamics of electrons in a band; motion of electrons in super-lattices, Semiconductors, Thermal properties of insulators; phonons, Landau levels - de Hass van alphen effect and integer quantum hall effect, Magnetism, Superconductivity — Introduction to Solid State Physics by C. Kittel, Solid State Physics by N. Ashcroft and N. D. Mermin — 8

-> P307 Nuclei and Particles — 8 — Nuclear systematics and stability (masses, sizes, spins, magnetic moments, quadrupole moments, energetics and stability against particle emission, beta decay), Nucleon-nucleon interaction, space-time symmetries, conservation laws, iso-spin symmetry, low energy interactions (effective range, shape independence, meson exchange picture), Liquid drop model, compound nucleus and fission, nuclear vibrations and rotations, Shell model, introduction to Hartree-Fock, spins and magnetic moments, Direct nuclear reactions, Mesons and baryons, resonances, SU(3) classification, iso-spin and strangeness, quark model, color, Weak interactions (nuclear and particle decays, neutrinos) — Introduction to Particle Physics by D. J. Perkins — 8

-> P401 Classical Mechanics-II: Mechanics of Continuous Media — 8 — Rigid body dynamics; Euler angle, Euler equations (should solve up to nutation of a top), Elastic Continua: Small deformations, stress tensor, elastic energy, equation of motion, Mechanics of continuous media, Strings: Euler Lagrange equation for continuous medium, Bernoullis and D'Alemberts solutions, Sturm-Liouville theory, Membranes: Scalar Helmholtz equation and its solution in variousgeometries, Fluids: Newtons second law for an ideal fluid, continuity equation, Euler equation, Bernoullis theorem, sound waves in fluids, Surface waves on Fluids: Tidal waves (long waves on shallow water), surface waves on deep water, solitary waves, Viscous Fluids: Viscous stress tensor, Navier Stokes equation, examples of incompressible flow, sound waves in viscous fluids — Classical Mechanics by N. C. Rana & P. S. Joag, Classical Mechanics by H. Goldstein, C. P. Poole, J. Safko — 10

-> P405 Atoms, Molecules and Radiation — 8 — Hydrogen atom including l.s coupling and hyperfine interaction, Helium atom introduction to exchange and correlation; variational calculation of ground and excitedstates, Introduction to the idea of effective potentials for electrons in many-electron atoms (Hartree theory and idea of self- consistency); use of Clementi-Roetti wave-functions, One-electron atomic systems in an electromagnetic field; dipole approximation and associated selection rules; Stark and Zeeman effect (note: instructor will have to introduce the students to time-dependent perturbation theory here), Einstein's A and B coefficients, population inversion, laser action, derivation of A

and B coefficients from semi-classical treatment of light-atom interaction, Molecular formation: Discussion of atom-atom interaction, van der Waals force, ionic interaction and covalent bond, Molecular structure: Hydrogen molecule MO and VB pictures; importance of correlations, Molecular spectra (restricted to two atom molecules) electronic, rotational and vibrational — Many Books— 8

-> P463 Astronomy and Astrophysics — 8 — Introduction: Astronomical objects, scales, distance ladder, astrometry, magnitude scale, Gravity - Kepler's law, Virial theorem, Radiation physics - radiative flux, transfer function, absorption, scattering and emission, Einstein coefficient, local thermodynamic equilibrium, source function and line formation, concept of opacities, Stars: Stars and stellar structures - stellar spectra, HR diagram, Equilibrium in stars, Star formation and Protostar, Stellar evolution, Supernovae, Black holes and gravitational waves, Interstellar medium, Galaxies: The Milky way Galaxy - distribution of matter, differential rotation, formation of the spiral arms, Elliptical and Spiral Galaxies, Evidence for dark matter, Active Galaxies - Active Galactic Nuclei, Seyfert Galaxies, Quasars, Blazers, Magnetic Fields: Astrophysical phenomena where magnetic fields are critical, Galactic magnetic fields - dust and synchrotron polarization, Faraday rotation, Zeeman measurements, Gravitational Lensing, Cosmology: Cosmological Observations and the Cosmological Principle, Newtonian Cosmology and Cosmological Models, Cosmic Microwave Background — Astrophysics for Physicists by Arnab Rai Choudhuri and Radiative processes in Astrophysics by Rybicki and Lightman — 9

-> P457 General Theory of Relativity and Cosmology — 8 — Review of Newtonian Mechanics. Special theory of relativity. prelude to general relativity, historical developments, 4-Vectors and 4-tensors, examples from physics, Principle of Equivalence, Equations of motion, gravitational force, Tensor analysis in Riemannian space, Effects of gravitation, Riemann-Christoffel curvature tensor, Ricci Tensor, Curvature Scalar, Einstein Field Equations, Experimental tests of GTR, Schwartzchild Solution, gravitational lensing, Gravitational waves: generation and detection, Energy, momentum and angular momentum in gravitation, Cosmological principle, Robertson-Walker metric, Redshifts, Big-Bang Hypothesis, CMB, Issues in Quantum gravity —A first course in General Relativity by Bernard Schutz, Gravity by James B. Hartle —8

-> P477 Special Topics in Quantum Mechanics — 8 — Group representation, Point group symmetry, Lie Groups; Schur lemma, orthogonality theorems, irreducible representations, accidental degeneracies; Irreducible tensor operators and direct product representations, Wigner Eckart theorem; characters of angular momentum states; SU(2), SU(3) representations, Path integrals, instantons, symmetries, conserved charges — Multiple Books — 9

-> P467 Nonlinear Dynamics, Chaos and Turbulence — 8 — General features of dynamical systems : Systems of differential equations with examples; control parameters; fixed points and their stability; phase space; linear stability analysis; numerical methods for nonlinear systems; properties of limit cycles; nonlinear oscillators and their applications; the impossibility of chaos in the phase plane; bifurcations: their classification and physical examples; spatial systems, pattern formation and the Turing mechanism; strange attractors and chaotic behaviour. The logistic map: Linear and quadratic maps; graphical analysis of the logistic map; linear stability analysis and the existence of 2-cycles; numerical analysis of the logistic map; chaotic behaviour and the determination of the Lyapunov exponent; universality and the Feigenbaum numbers; other examples of iterated maps. Hamiltonian Sytems: Phase space; Constants of motion and integrable Hamiltonians; Nonintegrable systems, the KAM theorem and period-doubling; applications. Fractal geometry: dimension of an object, Mandelbrot set, Julia set, iterated function systems. Spatio-temporal dynamics: Spatio-temporal chaos. Quantum Chaos: Quantum analogies to Chaotic behaviour, Correaltions in wave functions, chaos and Semi-classical approaches to Quantum mechanics — Nonlinear Dynamics and Chaos: With Applications in Physics, Biology, Chemistry and Engineering by S. H. Strogatz, Chaos and Nonlinear Dynamics by Robert C. Hilborn — 9

-> P475 Relativistic Nucleus-Nucleus Collision & Quark-Gluon Plasma — 8 — Introduction to high energy heavy ion collisions and Quark-Gluon-Plasma, comparison of big bang and the little bang, Thermodynamics: Relativistic gas (hadrons, quarks and gluons) and its statistical and thermodynamical properties, MIT Bag model, Hagedorn gas, phase diagram of QCD. Relativistic Kinematics: four vectors notation, rapidity variables, pseudo rapidity variables, light cone variables, relativistic invariants, Dalitz plot, cross sections. Collision Dynamics: initial state of nuclear collisions, fluid dynamical evolution, kinetic transport model, freeze-out and particle production. Experiments: a general overview of different experimental setup related to search for QGP and relevant observables. Signatures of QGP: collective flow, strangeness enhancement, jet quenching, electromagnetic probes, Hanbury-Brown-Twiss measurement — Hadrons and QGP by Letterssier and Rafelski, Introduction to High Energy Heavy Ion Collisions by C. Y. Wong — (Current Semester)

- **Physics Laboratory Courses:**

-> P245 Basic Electronics Theory and Laboratory — 5 — Foundations, passive elements, sources – dependent sources, Survey of network theorems and network analysis, Transient response of R-L circuit, R-C circuits, sinusoidal steady state response, Diodes and diode circuits, power supply – rectifiers, full wave rectifier without center tapped transformer, Bipolar junction transistors, constant current source, constant voltage source, field effect transistors, basic differential amplifier circuit — 10

-> P246 Advanced Electronics Theory and Laboratory — 5 — Operational amplifiers, feedback circuits and operations, Digital electronics, gates, universality of certain gates Boolean expressions, Other ways of realising logic functions, Multiplexers, flip-flops and latches, counters, sequential circuits – master slave flip-flop (S-R), edge triggered flip-flops, Transducers, signal averaging, lock-in amplifier, D/A & A/D converter, multi-channel analyzer, introduction to microcomputers and microprocessors — 9

-> P345 Optics Laboratory — 3 — Diffraction of laser light using single slit, thin wire and double slits, Diffraction by ultrasonic waves in liquids, Study of polarization of light, Splitting of Sodium D-lines using grating, Michelson Interferometer, Fabry-Perot Interferometer — 9

-> P343 Modern Physics Lab — 3 — Measurement of Elementary charge by Millikan oil drop method, Study of Zeeman effect, Study of Hydrogen Spectra (Balmer Series), Emission Spectra of Metals and Absorption Spectra in Iodine Vapour, Franck-Hertz Experiment using Neon Tube, Study of Electron spin resonance (ESR), Study of Magnetic Resonance with a Compass, Study of Lock-in amplifier — 10

-> P346 Computational Physics Laboratory — 6 — Introduction to Python, Representation of numbers on the computer, integers and floating point number, finite precision, Statistical description of data: Mean, Variance etc. Statistical inference, Error propagation, Curve fitting : Introduction to least squares, Straight line fitting, General linear and non-linear function fitting, Numerical Differentiation, Numerical Integration, Random number generators and random walk, Differential equations - Euler and Runge Kutta methods, Introduction to solving Partial Differential Equations, Finding roots of polynomials and transcendental equations, Minimisation of functions - golden section search, multivariable minimisation, gradient descent, conjugate gradient methods for quadratic and general functions, Solving system of linear equations using matrix algebra, Fast Fourier Transforms, Monte Carlo – Markov chain, Metropolis algorithm, Ising Model — Numerical Methods in Physics with Python by Alex Gezerlis — 7

-> P341 Nuclear physics and Instrumentation Lab — 6 — G.M. 1: Basics of GM Counter-Characteristics and Counting Statistics, G.M. 2: Application of GM Counter(Range of Beta Particles, Attenuation of Bremsstrahlung, Half-life measurement, Back Scattering), Study of Gamma Ray Spectroscopy by SCA and MCA, Study of Gamma-Gamma Coincidence using Na-22, Study of Rutherford Scattering, Construction of ADC and DAC circuits, Study of Compton Effect, Characteristics Response of a Photomultiplier Tube (PMT) — 10

-> P347 Solid State Physics Laboratory — 6 — Measurement of resistivity and determination of band gap using Four-Probe method, Bandgap measurement by Uv-Vis Spectroscopy and analysis of band structure, Hall effect of semiconductors Magnetoresistance and Hall effect of Bismuth Mono-atomic and diatomic lattice vibration study using electronic circuits, Study of phase transition in BaTiO3 I-V characteristics of solar cell C-V characteristics of solar cell Experiments with Seelab 3.0 — 10

- **Other Relevant External Courses:**

-> M206 Probability Theory — 8 — Combinatorial probability and urn models; Conditional probability and independence; Random variables – discrete and continuous; Expectations, variance and moments of random variables; Transformations of univariate, random variables; Jointly distributed random variables; Conditional expectation; Generating functions; Limit theorems; Simple symmetric random walk — S. Ross, “A First Course in Probability”, Pearson Education, 2012 — 8

-> Earth Structure and Dynamics — 8 — Mathematical treatment of deformation, Structural elements and their presentation, linear and planar features in rocks, Geometry and Formation mechanics of common deformation structures, Plate tectonics and the state of stress in the Lithosphere, Application of structural geology in understanding

planets, tidal forces and structures on extra-terrestrial bodies — Fundamentals of Structural Geology by Pollard and Fletcher — (Current Semester)

- **External/Self Read:**

– General Theory of Relativity and Cosmology books used; A first course in General Relativity by Bernard Schutz and Gravitation and Cosmology by Steven Weinberg
– Transiting Exoplanets book used Transiting Exoplanets by Carol Ann Haswell
– A First Course in Fourier Analysis book by David W. Kammler
– Magnetohydrodynamics and Plasma Physics course notes by Dr. Luke Chamandy and Astrophysical Magnetic Fields; From Galaxies to Early Universe by Anvar Shukurov and Kandaswamy Subramanian

Link to Physics Theory Courses: https://www.niser.ac.in/sps/course

Link to Physics Laboratory Courses: https://www.niser.ac.in/sps/tlm