Classifying non-redundancy in the HERA array

A thesis submitted in partial fulfilment of the requirements

for the degree of

Magister Scientae

in the

Department of Physics & Astronomy

The University of the Western Cape

UNIVERSITY *of the*
WESTERN CAPE

Author: Kabelo Malapane

(Student Number: 4052809)

Supervisor: Prof. Mario Santos

Co-Supervisor: Dr. Philip Bull

Declaration

I, ***Kabelo Malapane***, declare that this thesis, ***Classifying non-redundancy in the HERA array***, and the work presented in it are my own, that it has not been submitted before for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged as complete references.

Full Name: .....

Signature: .....

Date: .....

Acknowledgements

I would like to thank my supervisors Prof. Mario Santos and Dr. Philip Bull for their mentorship and guidance throughout the research project. I also like to thank Dr. Samir Choudhuri for taking the time to explaining and walking me through his simulations used in this study. In the same note I thank Dr. Piyanat (Boom) Kittiwisit for his insight on the project.

I would like to thank the National Research Foundation (NRF), South African Radio Astronomy Observatory (SARAO) and the Centre for Radio Cosmology (CRC) at the University of the Western Cape for providing the funding which enabled me to do this research project, and also providing the necessary equipments needed for me to complete my work.

I wish to thank the Inter-University Institute for Data Intensive Astronomy, IDIA and the Queen Mary University of London (QMUL) for providing the facilities used for simulating, storing and analysing the data used for this study. I would also like to thank the technical support staff that maintained the system for better accessibility and usage.

I also wish to thank my colleagues, the members of the CRC for making postgraduate experience a pleasant one and taking your times to organise the weekly meetings which have been quite insightful and has made these Covid-19 times feel less suffocating. Its been an honor and privilege to know and engage with you all.

Finally to my family, I would like to wholeheartedly thank my mother Jane Thibela for supporting me throughout my entire life and always believing that I can achieve anything I put your mind to. To my sister Cyncinatia who has advised me on every aspect of life, especially on academics being a student herself. To my two big brothers who have been a source of encouragement and support, thank you very much.

Abstract

Classifying non-redundancy in the HERA array

Kabelo Malapane

M.Sc. Thesis

Department of Physics & Astronomy

The University of the Western Cape

HERA is a highly redundant radio interferometer array, where pairs of receivers with the same position vector between them should see exactly the same signal from the sky. We can use this fact to do a really good job of calibrating them. Unfortunately, the receivers are not perfectly identical, and so they don't see exactly the same signal. This is called "non-redundancy". This project classifies the level of redundancy using a clustering machine learning technique. The aim is to see if any particular clustering algorithm can group different segments of the array into very similar blocks, so we can at least do a good job of redundantly calibrating within those blocks. We call this new calibration method, logi_cal, while the standard calibration method used in HERA is called redcal.

We simulated six cases of non-redundancy for a 124 antenna array, and the results show that for these six cases, the calibrated antenna gain solutions when using logi_cal have improved compared to the ones for redcal. For the case where we stretch the primary beam, we improved the calibration by 12.2%. When we perturb the side lobe, we get 13.6%, and for a case where the primary beam is rotated, the improvement is about $6.8 \pm 0.1\%$. Logi_cal works for primary beam non-redundancies and does not improve the gain solutions for positional non-redundancies in the system.

Keywords: Non-redundancy, low frequency arrays, epoch of reionisation, Clustering

September 16, 2022

Contents

List of Figures

List of Tables

1 Introduction

The 21 cm signal emitted by neutral hydrogen could reveal the secrets of the early Universe and give us an accurate look into how the first stars and galaxies formed, and how the Universe became reionized in the so-called Epoch of Reionization (EoR). Several experiments have been designed to explore this era by measuring the 21 cm signal to study the EoR (e.g. (Zaroubi, 2010; Parsons et al., 2010; Deboer et al., 2017; Lonsdale et al., 2009)). To measure the 21cm signal, precision calibration is key in separating the neutral hydrogen signal from bright astrophysical foregrounds (Dillon & Parsons, 2016), and a highly sensitive instrument is needed to observe this signal. The Hydrogen Epoch of Reionisation Array (HERA) is such an instrument.

HERA uses redundancy within the array to calibrate the data. By redundancy, we mean when radio interferometer arrays are arranged into a regular pattern, pairs of receivers with the same position vector between them (i.e. the same distance and orientation) should see exactly the same signal from the sky. We will use simulations of the HERA observation pipeline to train and test calibration algorithms to see how well they perform. We will ultimately apply such algorithms to the real data from the HERA array to classify the level of non-redundancy and compare it to what is currently known.

1.1 Epoch of Reionisation

The evolution of the Universe, and in particular the evolution of hydrogen in the Universe from the Big Bang to the present, has been of particular interest to cosmologists. After the Big Bang occurred, the Universe was in a hot, fully ionised state where the neutral fraction was essentially zero. A few hundred thousand years later, the Universe had expanded and cooled, so the hot plasma which the Universe was composed from became electrically neutral, via the recombination process (the formation of neutral atoms from hydrogen ions and electrons). The cosmic microwave background (CMB) then free-streamed through the Universe. the Universe stayed in a fairly neutral state for about a billion years after this. Over time, areas of higher gas density began to collapse under gravity, and the neutral matter in the Universe began to clump together. Eventually, the first stars, black holes and galaxies formed. The formation of these structures reionises the gas in the intergalactic

medium (IGM), leading to the fully ionised gas between galaxies that we see today.

This period of time is called the Epoch of Reionisation (EoR). Figure 1.1 shows the process of what is believed to be the evolution of the Universe. The yellow line on the figure shows the span of the EoR, which is observable in the redshift range of $6 \lesssim z \lesssim 15$ and occurs for about 0.6 billion (between 400 million and 1 billion) years (Madau et al., 1997; Thomas et al., 2009).

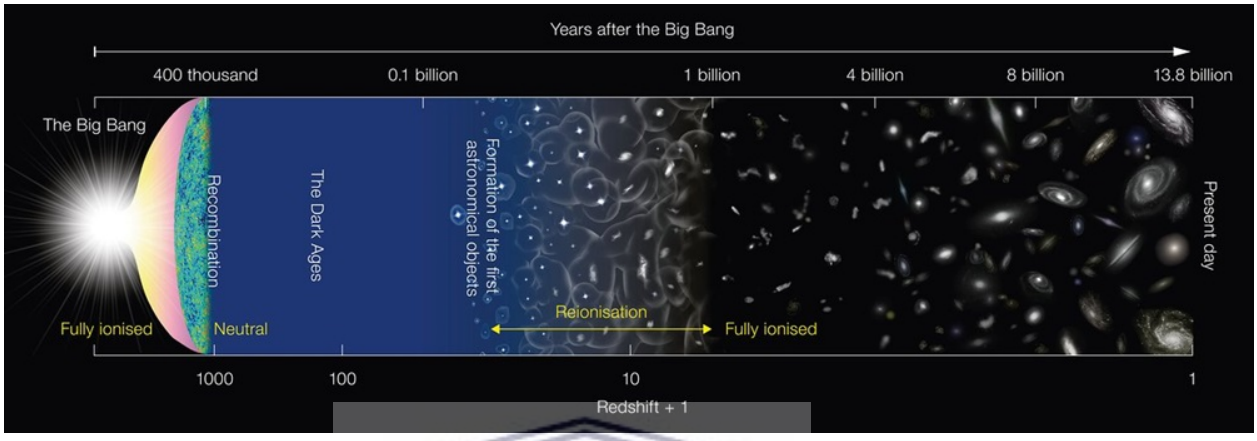


Figure 1.1: A map showing the history of the Universe, including the shift from neutral to ionized hydrogen, resulting in the Universe we see today. The upper axis shows the time that elapsed since the Big Bang in years, and the lower axis shows time in redshift (1 + z). The yellow double-sided arrow shows the epoch of reionisation timeline (Kwon, 2017).

It is difficult to learn about the Universe in this era, given the limited light emitted by the young stars and galaxies. However, new state-of-the-art telescopes are to be deployed like the James Webb Space Telescope (JWST) to study the first light, reionization and galaxy assembly (Windarto, 2017). Thanks to the quantum mechanical transition between the ultrafine energy levels of hydrogen (the so-called spin-flip transition), there is radio emission with a wavelength of about 21 cm. When the radiation reaches the Earth, a 21 cm line for the EoR is observed at a frequency of about $100 - 200$ MHz, corresponding to a redshift range of $z \sim 6 - 12$ (Bernardi et al., 2009; Zaroubi, 2013; Deboer et al., 2017).

1.1.1 Astrophysical foregrounds

Observing the 21 cm line during the EoR is plagued by astrophysical foregrounds and other non-astrophysical interferences (which include instrumental response, and

ionospheric disturbances, to name a few). These astrophysical foregrounds are $\sim 10^5$ orders of magnitude brighter than the 21 cm signal (Morales & Hewitt, 2004; Santos et al., 2005; Furlanetto et al., 2006; de Oliveira-Costa et al., 2008). Of course, the signals from these foregrounds can themselves have considerable scientific interest, like galaxy surveys (e.g. (Hurley-Walker et al., 2017, 2019)). However, for the EoR experiments, they are considered contaminants, which need to be separated from the 21 cm cosmological signal.

Here we mention the astrophysical foregrounds that are major contributors to the contamination of the 21 cm signal. These include the Galactic synchrotron emission, Galactic free–free emission and the extragalactic emission (including extragalactic point sources). These contribute about $\sim 70\%$, $\sim 1\%$ and $\sim 27\%$ respectively (Shaver et al., 1999).

- Galactic synchrotron emission: The result of cosmic-ray electrons and positrons propagating in the interstellar magnetic field. Electrons move at an angle to the magnetic field and feel the Lorentz force; therefore are accelerated and radiate in a cone-shaped beam. Their combined spectra mostly follow the power laws. This emission is brightest in the galactic plane (from the milky way) and includes other galaxies with any ongoing star formation or accretion activity (Jarvis et al., 2015).

- Galactic free–free emission: Also known as bremsstrahlung (German: "braking radiation"), from electrons scattering by ions in a very hot plasma. It is the least well-known of the three diffuse Galactic emissions, which dominate the mm and cm wavelength sky (Smoot, 1998). It results from the deceleration of a charged particle when deflected by another charged particle, typically an electron with an atomic nucleus. As the name suggests, an electron starts in a free (unbound) state and ends in an unbound state. This type of emission is mostly found in galaxy clusters.

- Extragalactic point sources: Includes emission from sources outside the Milky Way, from extragalactic radio and mm/sub-mm sources, with radio galaxies being the main sources. These include the synchrotron and free-free emission from other galaxies.

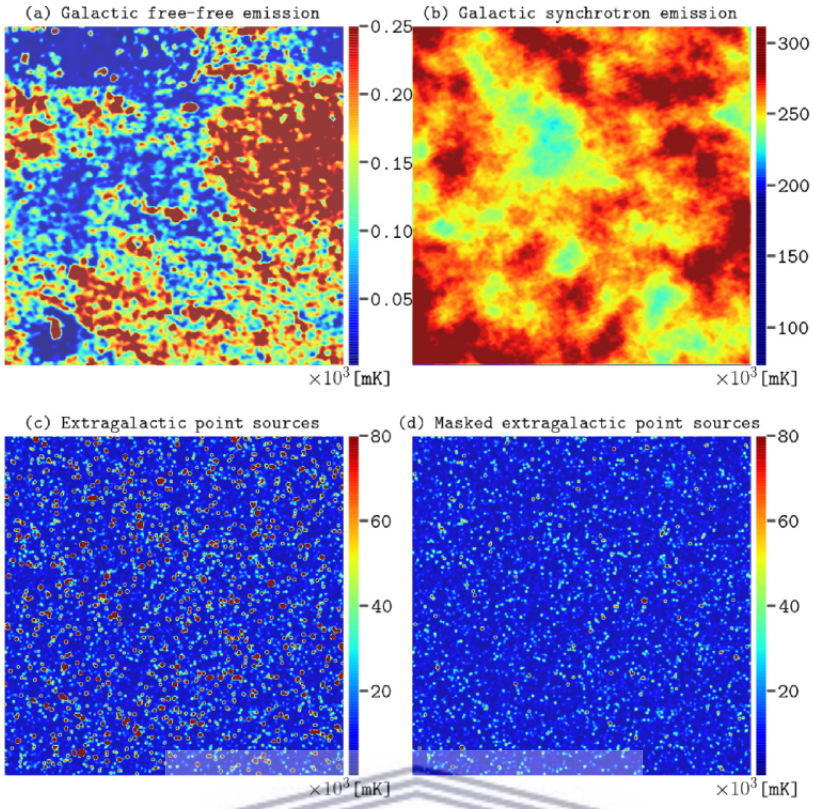


Figure 1.2: A 10°×10° sky region showing maps of foreground emissions at 150 MHz from the (a) Galactic free–free emission, (b) Galactic synchrotron emission, (c) Extragalactic point sources, and (d) Masked extragalactic point sources (i.e. the case with the brightest point sources removed). Every color bar is in units of $\times 10^3$ *mK*. With (d), the brightest point sources are removed from (c), leaving a background of undetected point sources that still contaminate. Credit: (Lian et al., 2020).

These plots show that the Galactic synchrotron emission is the major contributor, compared to the other two diffuse Galactic emissions, to the foreground problem. They also show that the foregrounds are $\sim 10^4 - 10^5$ orders of magnitude brighter than the 21 cm signal with a brightness temperature of a milli-Kelvin (Fialkov & Loeb, 2013).

1.2 21 cm Signal

Studying and imaging the Universe and its progression to the current day remains one of the exciting challenges facing modern cosmology. Hydrogen is the most abundant element in the Universe, so finding a way to trace this hydrogen throughout

time can give us a way of observing the Universe. Over the years, the observation of the 21 cm line of neutral hydrogen (HI, pronounced "H one") has enabled the mapping of the Universe in three dimensions, with redshift providing line-of-sight distance information (e.g. (Pritchard & Loeb, 2012; Cisewski et al., 2014)). However, the first HI galaxy surveys were done way before. For example, in the late 1970s, Vera Rubin used in her discovery of dark matter, confirming Fritz Zwicky's 1933 prediction of dark matter (Trimble, 1987; De Swart et al., 2017). The mapping of the 21 cm line is also helped by the fact that at about 1420 MHz, this radiation from hydrogen can pass through dust clouds and gives us a more complete map of the hydrogen than all the optical observations.

In radio astronomy, observations of the 21 cm line have been crucial in determining the relative speed of each arm of our galaxy. It is done by using the fact that our galaxy is distributed throughout with hydrogen, therefore by calculating the Doppler shift of each of these lines, one can calculate the speed. It has also helped in the calculation of the rotation curve of our galaxy, thereby allowing us to use a plot of the rotation curve to determine the distance to certain points within the galaxy. This scientific approach has also extended to other astronomical observations. The rotation curves of the 21 cm line are often used to track the dynamics of galaxies. While the traditional method of observing 21 cm emission detected lines only in relatively localized galaxies, the 21 cm line has appeared in absorption against a background of noisy radio waves, that is, from background sources for individual systems in redshifts $z \lesssim 3$ (Kanekar et al., 2007).

The 21 cm line was theoretically predicted in 1942 by Dutch astronomer H. C. van de Hulst (van de Hulst, 1945) and was first observed in 1951 by Harold Ewen and Edward M. Purcell (Ewen & Purcell, 1951) at Harvard, and then later by other observers in Holland and Australia. In radio astronomy, the telescopes look for radiation from the cold hydrogen gas inside galaxies. Because of the line's narrowness with a well-measured rest-frame frequency, it can be used as a probe for gas velocity distributions in the Milky Way and other nearby galaxies in local space. The 21 cm line has been theorized to be a crucial catalyst in the continued study of the Big Bang, from recombination to reionization. This line has two applications. First, by mapping the intensity of the redshifted 21 cm of radiation, in principle, a very accurate image of the matter power spectrum at the time after recombination can be provided (e.g. (Harker et al., 2010; Liu & Tegmark, 2011; Parsons et al.,

2012)). Second, the radiation-ionized neutral hydrogen from stars and quasars appears as holes in the 21 cm background, indicating how the Universe was reionised e.g. (Natarajan & Yoshida, 2014).

In quantum mechanics, the 21 cm line can be absorbed or emitted by a neutral hydrogen atom, and in doing so, we have a way of seeing hydrogen. The 1420 MHz radiation originates from the transition between the two levels of hydrogen's ground state F=0 and is slightly split (known as hyperfine structure) by the interaction between electron and nuclear spins. Due to the quantum properties of radiation, hydrogen absorbs 1420 MHz in the lower state, and observations of 1420 MHz of emission signify a previous excitation to the upper state F=1. The transition between states is visualised as spin-flip, where the "spin" is not a literal classical spinning charge sphere, so there is no clear classic analogy, it is merely a description of the behaviour of the angular momentum in quantum mechanics. A more scientific explanation is that it's going between the symmetric and anti-symmetric states. This process is shown in Figure 1.3.

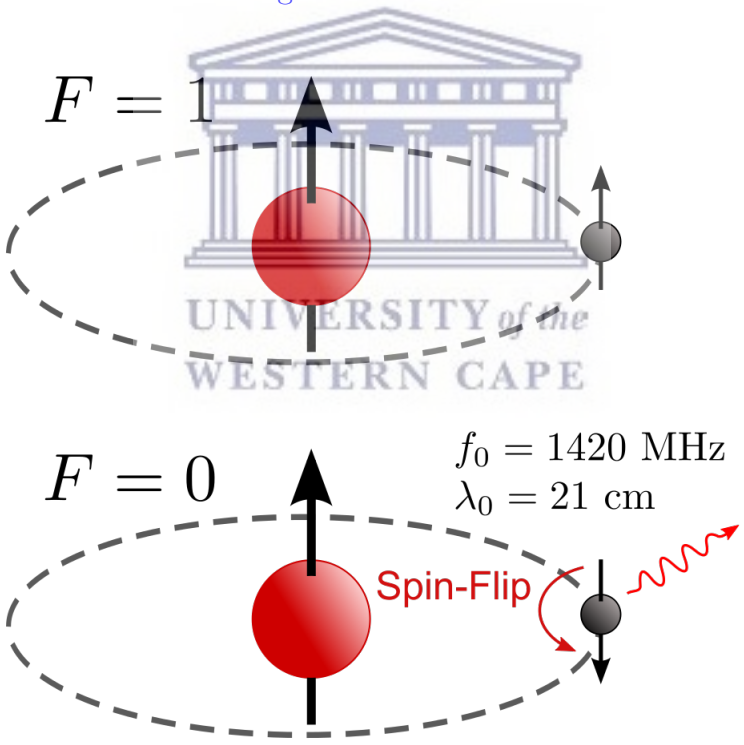


Figure 1.3: The transition of the Hydrogen 21-centimeter line (Tiltec, 2021)

There are other hyperfine transitions that may be useful in probing cosmology: (a) The 8.7 GHz hyperfine transition of $^3He+$, for probing Helium reionisation (Bagla & Loeb, 2009), and (b) the analogue of the 21 cm line, the 92 cm deuterium (Sig-

urdson & Furlanetto, 2006). However, since these elements are not as abundant as neutral hydrogen, it is less desirable to use them for probing cosmology. To do 21 cm cosmology, it is important to consider what's called the cosmic microwave background (CMB, CMBR), electromagnetic radiation left over from an early stage of the Universe (Sunyaev, 1974). The radiation from the CMB is almost isotropic and is radiating the strongest in the microwave region of the radio spectrum, filling all space. So since the CMB is used as a backlight, what we measure is the brightness temperature contrast with the 21 cm line (Pritchard & Loeb, 2012).

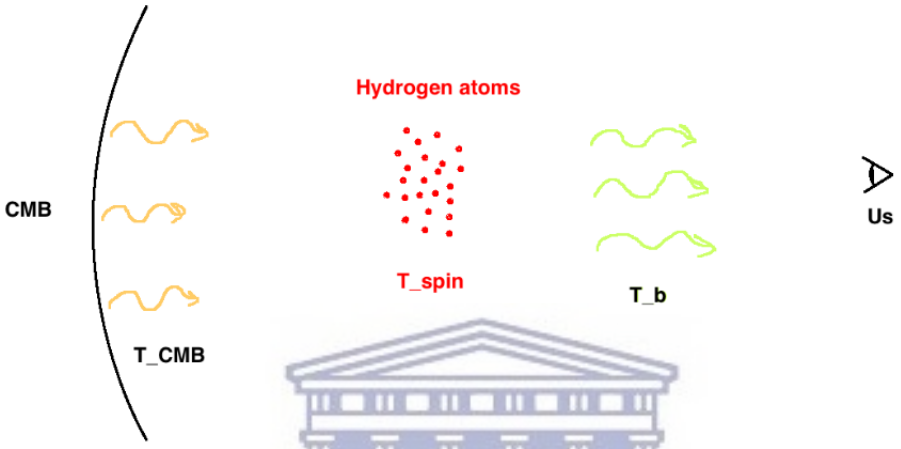


Figure 1.4: This is an illustration of a set up of the various components relevant to radiative transfer. We have some signal coming from the CMB with temperature $T_{CMB} \equiv T_\gamma$ interacting with a cloud of hydrogen atoms with spin temperature T_S, which then results in a brightness temperature T_b that is then measured by the radio telescopes. Credit: (Day, 2015)

This brightness temperature contrast corresponds to the absorption or emission of the hydrogen atoms. Then, that absorption or emission is going to be proportional to the hydrogen density δ, that is, the more hydrogen there is, the more the absorption or emission. It will also depend on the neutral fraction χ_{HI}, which indicates what fraction of the atoms are neutral. The relative population of the spin states define the spin temperature, T_S, through the relation,

$$\left(\frac{n_1}{n_0}\right) = \left(\frac{g_1}{g_0}\right) exp\left\{\frac{-T_*}{T_S}\right\}, \tag{1.1}$$

where g_1 and g_2 are the spin degeneracy factors and $T_* \equiv h\nu_0/k_B \equiv E_{10}/k_B = 68mK$ is equivalent to the transitional energy E_{10}. The detailed 21 cm line signal

depends on the transmission of radiation through the gas along the line of sight, the peculiar velocity term ($dv_\parallel/dr_\parallel$), and the background radio sources, usually the CMB with temperature T_γ. According to the derivation from Furlanetto et al. (2006); Furlanetto (2016), the observed brightness temperature T_b due to the 21cm line at a frequency ν is given by

$$T_b(\nu) \approx \frac{T_S - T_\gamma(z)}{1+z}\tau_{\nu_0} \tag{1.2}$$

$$\approx 9\chi_{HI}(1+\delta)(1+z)^{1/2}\left[1-\frac{T_\gamma(z)}{T_S}\right]\left[\frac{H(z)(1+z)}{dv_\parallel/dr_\parallel}\right]\ mK, \tag{1.3}$$

Figure 1.4 in conjunction with Equation 1.3 shows that if $T_S < T_\gamma$ then $T_b < 0$ which yields an absorption signal. It yields an emission signal when $T_S > T_\gamma$. Both these regimes are important for the high redshift Universe however, the general agreement is that currently, during the reionization era, emission will dominate.

2 Introduction To Radio Astronomy

Radio astronomy is a sub-field of astronomy that deals with observations done at radio frequencies. The first astronomical radio waves were detected by Karl Jansky at Bell Telephone Laboratories, as he reported radiation coming from the Milky Way (Jansky, 1932, 1933). This observation opened new possibilities in the field of astronomy, allowing not only the observation of stars and galaxies, but also entirely new classes of celestial objects, such as radio galaxies, quasars, pulsars, and masers (Noordermeer et al., 2005). Early studies in observational astronomy were limited to visible objects that lie in the optical range of the electromagnetic spectrum, that is, anything that can be observed by the human eye. However, with the introduction of radio observations, things we could not see before with our eyes now became visible, like radio jets, which are material spewing from the centres of some galaxies at close to the speed of light and emitting strong radio waves.

Observations using radio waves not only reveal new structures from the celestial objects that were not visible in the optical, but these waves can also penetrate through cosmic dust in the interstellar medium. This ability to look through dust makes observations that could not be done in optical possible, leading to major discoveries of the 21 cm hydrogen line and the cosmic microwave background (Penzias & Wilson, 1965), which has become a staple in studying the history of the Universe.

Because of the Earth's atmosphere, only radio and optical/near-infrared observations can be made on the ground. This is due to the Earth's atmosphere absorbing most of the infrared (IR), ultraviolet (UV), X-ray, and gamma-ray wavelengths, as shown in Figure 2.1. The figure shows how narrow the optical range is, which in turn limits the number of astronomical objects that can be studied. Because the radio window is so broad, it includes a wide range of astronomical sources, thermal and non-thermal radiation mechanisms, and propagation phenomena, which can all be observed at radio wavelengths. With the number of discoveries made using optical observations despite such a narrow window, the prospect of new scientific knowledge that can be acquired with radio observation with such a broader window has made a wide variety of radio telescopes and observing techniques necessary to cover the radio window effectively.

Radio telescopes are built in different sizes and shapes, but often they are large parabolic ("dish") antennas. These are specialised antennas and receivers built to

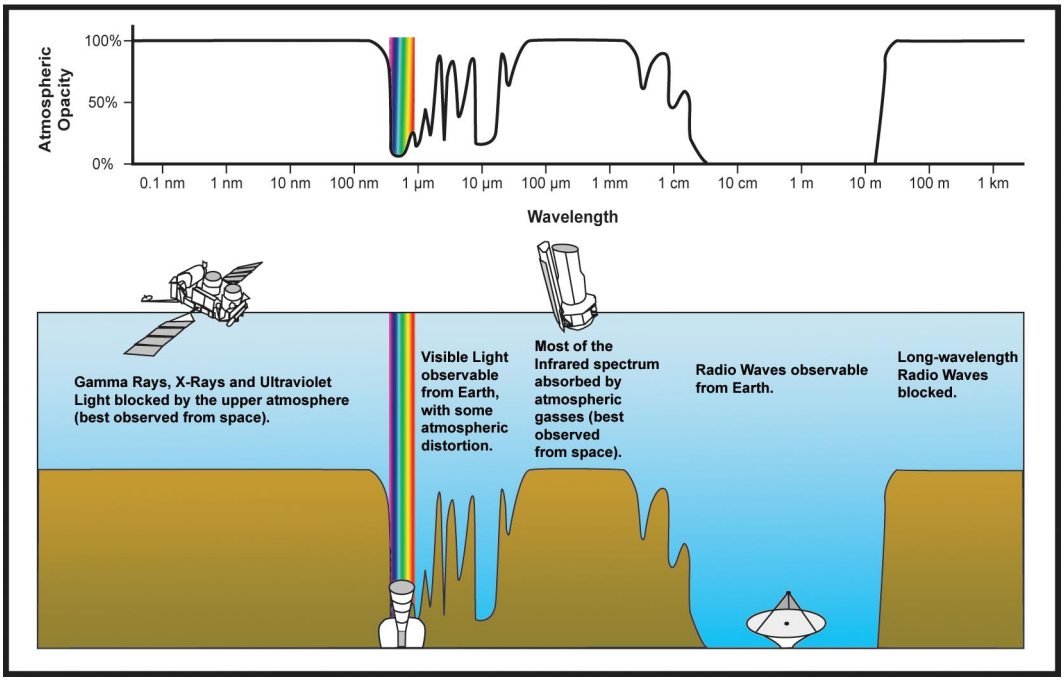


Figure 2.1: This figure shows different types of astronomical observations could be made at a given wavelength or frequency ranges. When plotted on logarithmic wavelength or frequency scales the visible window is much narrower than the radio window when plotted on logarithmic wavelength or frequency scales, indicating that in comparison, radio astronomy includes a lot of astronomical sources and emission mechanisms. Credit: NASA/IPAC.

detect radio waves coming from astronomical radio sources (Marr et al., 2015). Their shapes are influenced by which part of the radio window they are trying to probe. Radio waves come from far away sources like galaxies, so their signals are very weak, requiring large parabolic dishes to collect as much information as needed. These radio telescopes can either be operated individually, or they can be linked together electronically in an array. These linked antennas can be two or more, separated by a small distance or thousands of kilometres apart.

2.1 Interferometer

The groundwork for interferometry in astronomy dates back to the optical work of Michelson (Michelson, 1890, 1920), and Michelson and Pease (Michelson & Pease, 1921), who were able to measure the diameters of some of the nearer and larger stars such as Arcturus and Betelgeuse (0.047 arcseconds). Early on, radio astronomers recognized the fundamental similarity of the theory of optical and radio radiation

fields, and they used the theory behind these optical experiments to build the theory of radio interferometry (Thompson et al., 2017). The idea of interferometry is to use two or more telescopes to detect an electromagnetic wave from a source and combine them. So this means that waves from the same source only combine constructively when they are in phase and this is related to the distance travelled by light between the light source and the telescope elements.

The angular resolution for most telescopes of diameter D, observing signals of wavelength λ is given by:

$$\theta \approx \frac{\lambda}{D} \tag{2.1}$$

From the equation, the angular resolution is highly dependent on the size of the telescope, that is, the bigger the telescope, the more resolved the sources are. However, there is a limit to how big we could make a single dish and still be able to operate it. As an example, the Green Bank Telescope (GBT) is the world's largest fully steerable radio telescope operating between wavelengths ranges of 2.6 mm and 3 m (0.1–116 GHz) and $D = 100$ m are limited to $\theta \gg 1$ $arcsec$, still given its size it is not large enough to achieve sub-arcsecond resolution at radio wavelengths (Condon & Ransom, 2016). To overcome this limitation, radio astronomers use multiple radio telescopes at the same time, and this technique is called interferometry.

In interferometry, the angular resolutions are not calculated using the diameter of the individual antennas, but by the distance between the two farthest antennas, called a baseline. This baseline is the vector connecting any two antenna elements in an array. So with interferometry, they are effectively creating a single telescope, improving the resolution of the telescope, but without increasing the light-gathering power. A signal from a radio source arrives at each antenna at a slightly different time (due to different travel lengths) depending on the location of the antenna in the array. In Very-Long-Baseline Interferometry (VLBI), each antenna has a clock to record the observation time of the source, then the signals are later combined with every other antenna in an array, taking into account the time delay (τ_{ij}) using computer software. However, for telescopes like MeerKAT, signals are combined in real-time as they arrive, not saved for later. This process is shown in Figure 2.2:

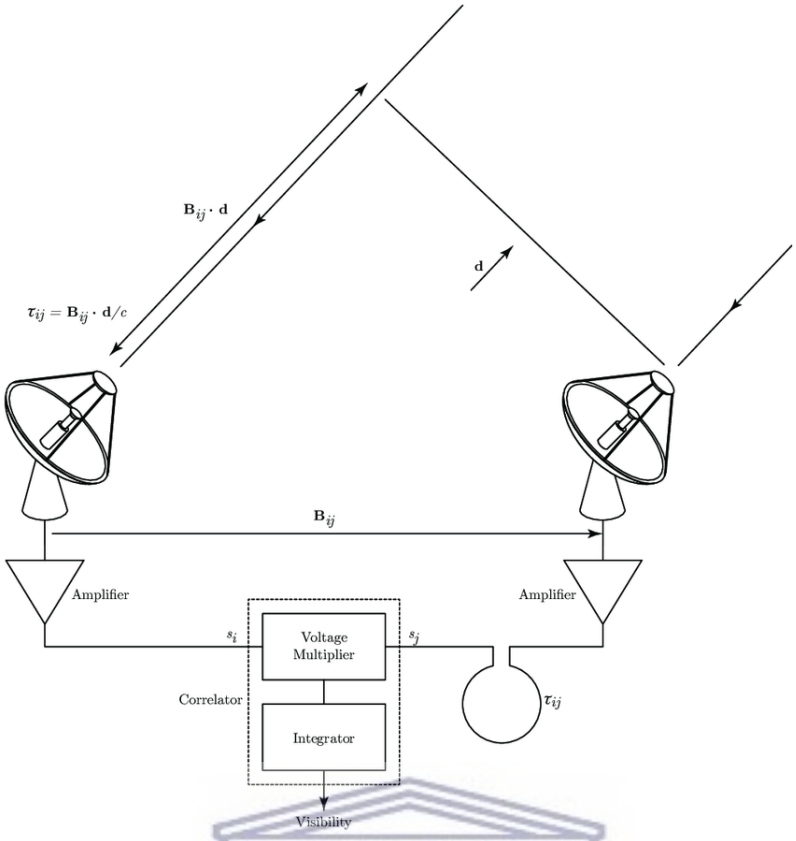


Figure 2.2: This figure shows a two-element interferometer, Credit: (Mahmoud et al., 2011). It shows antennas i and j, separated by baseline vector B_{ij}.

To be more specific, we consider a simple interferometer model and assume a quasi-monochromatic wave with identical elements and perfect electronics. A signal from a source arrives at antenna-elements/receivers i and j separated by baseline vector B_{ij}. This signal arrives at an angle ϕ to the normal of the baseline, from direction d reaching each antenna receiver element i, j at slightly different times resulting in a phase difference. The signal is detected by a current induced in an antenna receiver system, where they can be measured as a voltage $s_i(t)$ at receiver i that is sampled and digitized at regular times t. The pair of receivers shown in Figure 2.2 can be used as an interferometer to measure the difference in phase between the signals s_i and s_j due to the time delay between the received signals defined as:

$$\tau_{ij} = B_{ij} \cdot \frac{\vec{d}}{c} \quad , \tag{2.2}$$

where c is the speed of light. This time delay τ_{ij} can be roughly approximated using the positioning of the antennae relative to the source direction (provided by an Ephemerides service). The Ephemerides service gives the trajectory of naturally occurring astronomical objects as well as artificial satellites in the sky, i.e., the position and, if possible, the velocity over time. τ_{ij} can also be precisely determined by the resulting interference pattern in the cross-correlation between the signals:

$$(s_i \cdot s_j)(\tau) = \int_{-\infty}^{\infty} \overline{s_i(t)} s_j(t+\tau) dt \tag{2.3}$$

In radio astronomy, we use the coordinate system (u,v,w), where w is along a reference direction to the phase centre and (u,v) are in the orthogonal plane, with u East-West and v North-South (the u-v plane). So unlike a single dish telescope that measures the intensity, what an interferometer measures is called coherence. The data from these interferometers is called a visibility, given by $V_{ij} = (s_i \cdot s_j)(\tau_{ij})$. Put simply, the visibility is the amplitude and phase information of the cross-correlated signals between pairs of antennas. The complex visibility is related to the 2D Fourier transform of the emission on the sky $T(l, m)$, over the projection of the celestial sphere in the l-m plane onto the u-v plane defined as,

$$T(l,m) = \int \int V(u,v) e^{-2\pi i(ul+vm)} du dv. \tag{2.4}$$

In terms of the sky brightness distribution, the complex visibility and Fourier Transform relation in the u-v plane can be written as:

$$V(u,v) = \int \int T(l,m) e^{-2\pi i(ul+vm)} dl dm. \tag{2.5}$$

A radio telescope consists of a number of antennas, with the baselines between them representing measurement points in u,v space. Then this means we will need to convert an array of dishes on the ground to a set of points in u,v space. Note that a consistent coordinate system needs to be established between the u,v space and the baselines, where antenna separations are typically measured in units such as meters along the ground. According to the note by Gary (2019), the complex

visibilities of a source measured by the interferometer is defined as

$$V(u,v) = \int A(l,m)I(l,m)e^{-i2\pi(ul+vm)}dldm \tag{2.6}$$

where $V(u,v)$ is the true visibility evaluated at u, v,
(u,v) are the projected baseline coordinates in wavelengths, $u = B_{l,\lambda},\ v = B_{m,\lambda}$,
$A(l,m)$ is the normalized primary beam pattern (beam of a single antenna),
(l,m) are the direction cosines with respect to the phase center, and
$I(l,m)$ is the brightness distribution of the source

Most radio telescopes have been built with more than just two antennas, so an array of n antennae has $\frac{n(n-1)}{2}$ baselines (one per pair of antennae) and so $\frac{n(n-1)}{2}$ visibilities can be obtained. However, each baseline changes over time due to the Earth's rotation, in the case where readings are taken over intervals of hours. For these long observational periods, the amount of data and computational power it takes to process the signals measured from all antennas is an enormous challenge in this field. Take HERA as an example, which will eventually have about 350 antennae dishes and will produce about 260 terabytes (TB) of raw data for a typical observation period of 12 hours (La Plante et al., 2021). In a night, the real-data volume recorded by HERA will be well over 50 TB when using baseline dependent averaging, as described in (Wijnholds et al., 2018). With the amount of data being produced and the large number of arrays deployed, good data quality needs to be assured so good calibration methods are needed.

2.2 Experiments for 21 cm observations

The importance of 21 cm observations in mapping the high-redshift universe using redshifted radio emission from neutral hydrogen has led to the design of several experiments to probe the 21 cm line. Due to how faint the 21 cm signal is, these experiments have to fit certain criteria in their design: they must have low noise, a wide field of view, and moderate angular resolution. We focus mainly on broadband interferometers instead of single-dish telescopes that were built for general-purpose observations, like the Green Bank Telescope (GBT) (Prestage et al., 2009) and the Parkes radio telescope (Staveley-Smith et al., 1996). In this chapter we discuss some of the experiments ongoing for observing the 21 cm line, namely LOFAR (e.g. (van

Haarlem et al., 2013)), MWA (e.g. (Lonsdale et al., 2009)), PAPER (Parsons et al., 2010), as well as the ones that are planned and under construction, the HERA Deboer et al. (2017) and the SKA (e.g. (Koopmans et al., 2015)).

The Low-Frequency Array (LOFAR): Currently the largest radio telescope located in the Netherlands, operating at the lowest frequencies. It is a multipurpose sensor network built to handle extremely large data volumes with its innovative network infrastructure and computers. It is an interferometric array of radio telescopes that at present, is using about 20,000 small antennas concentrated in 52 stations. LOFAR focuses on the EoR in the window ($6 < z < 10$), exploring large areas of the sky with low-frequency radio waves ($1.5 < z < 7$), and constantly monitoring radio wave transients from some of the most energetic explosions in the universe. LOFAR is observing the largely unexplored low-frequency range from 10–240 MHz (van Haarlem et al., 2013).

Figure 2.3: An aerial image of the LOFAR array, from August 2011 (Haarlem et al., 2013).

The Murchison Widefield Array (MWA): A low-frequency radio telescope located in Western Australia, consisting of 4096 spider-like antennas that are arranged in 256 regular grids called 'tiles', spread over several kilometres. It is operating in a

wide frequency range (70–300 MHz) and its extreme (digital) pointing agility allows for flexible tuning. The telescope was built to answer four key scientific objectives: how does the Sun affect the environment on Earth; time-domain astrophysics; what more could we learn from the Milky Way galaxy when looking at its magnetic field, and pulsing and exploding stellar objects; and the pursuit of the intergalactic hydrogen gas that surrounded early galaxies during the EoR. The instrument has 3 bands of observations for the 21 cm experiment (Jacobs et al., 2016): ultralow-band (75–100 MHz), low-band (139–167 MHz), and high-band (167–197 MHz), with low- and high-bands accounting for more than 90% of the observed data.

Since its operation, the MWA is and has performed large surveys of the entire Southern Hemisphere sky. Since one of its main science goals is to probe the EoR, the MWA is pioneering the testing and development of new techniques for the separation of cosmological foregrounds for the 21 cm signal (Lonsdale et al., 2009).

Figure 2.4: Image of the MWA array with its spider-like antennas. Credit: Greg Rowbotham ICRAR UWA 2016

The Precision Array for Probing the Epoch of Reionization (PAPER): A low-frequency radio interferometer that was located in South Africa, built to detect 21 cm hydrogen (HI) fluctuations occurring when the first galaxies ionized intergalactic gas around 500 million years after the Big Bang. PAPER mapped the intensity of hydrogen emission at 21 cm in a high redshift range ($7 < z < 12$). It had a

32-antenna array at the NRAO site and a 64-antenna array in the Karoo reserve in South Africa, with plans to expand to 128 antennas. The PAPER project took an incremental engineering approach, optimizing each component in the array in a staged process to minimize potential problems with subsequent calibration and analysis. This was done to tackle the foreground problem, given that they are 5 orders of magnitude brighter than the background for detecting reionization (Parsons et al., 2010). PAPER, along with the MWA, are frontier projects within the Hydrogen Epoch of Reionization Array (HERA) program. Since PAPER is no longer functioning, its receivers were used on HERA dishes during HERA's first observational campaign.

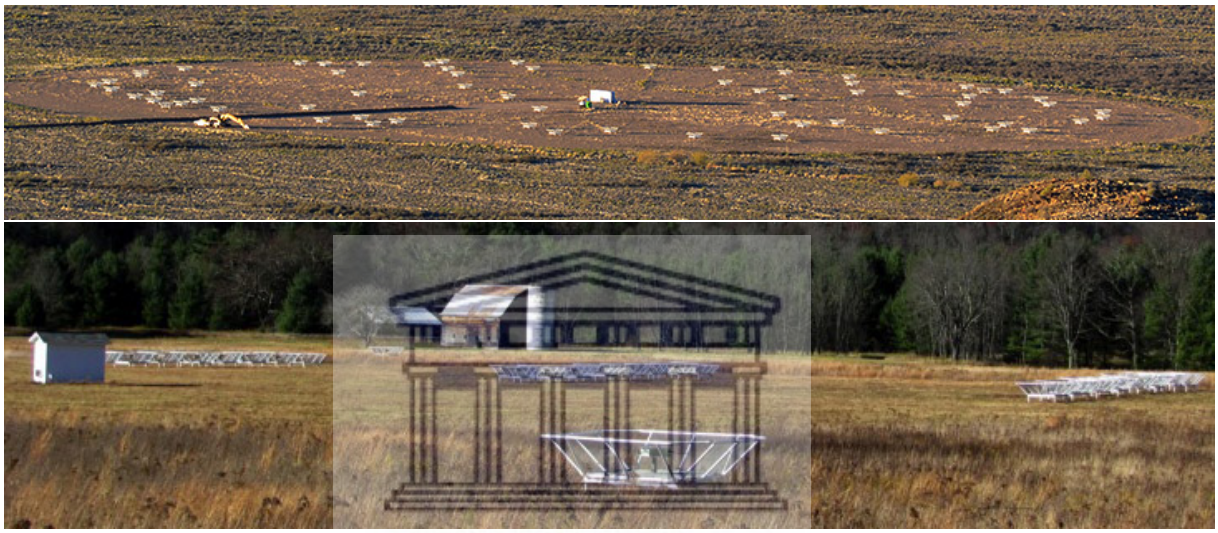


Figure 2.5: PAPER Green Bank with a 32 antennas configuration optimized for power spectrum sensitivity (Bottom) , PAPER South Africa (64 antennas) in an imaging configuration (Top). Credit (website): (PAPER, 2012).

For this project, we are focusing on the Hydrogen Epoch of Reionization Array (HERA) radio telescope. HERA is currently the most sensitive 21cm EoR experiment in the world. It is specifically designed to detect fluctuations in the emission from neutral hydrogen gas found throughout the Universe during the Epoch of Reionisation and Cosmic Dawn, when stars, galaxies and black holes formed. It is an interferometer located in South Africa that, when fully complete, will consist of 350 elements consisting of 14-m parabolic dishes, and it will detect radio waves in the low-frequency range of 50–250 MHz (Deboer et al., 2017).

Figure 2.6: Rendering of the 320-element core (left) of the full HERA-350 array and the picture on the right we have on the foreground of 19 HERA 14 m, zenith-pointing dishes (with PAPER elements in the background) currently deployed in South Africa (right) (Deboer et al., 2017)

The Square Kilometer Array (SKA): An international project to build the world's largest radio telescope. SKA has four precursor facilities that are already operating. Two of them have been mentioned: HERA and MWA; and the other two, the Australian SKA Pathfinder (ASKAP) and MeerKat in South Africa. SKA-Mid (mid-frequency) will be located in Africa, hosting a mid-frequency dish array, and SKA's low-frequency telescope will be in Australia, hosting a low-frequency aperture array antenna. The SKA telescope is being built in phases, SKA1 and SKA2. SKA1 accounts for only 10% of the capability of the whole telescope. SKA-Mid will have the 64 MeerKat dishes with 133 additional antennas totalling nearly 200, located in South Africa. SKA-Low will have more than 130,000 antennas in total built in Australia. To break it down, it will have 512 stations (each with 256 individual antennas) arranged in a large core with three spiral arms, spread over a distance of 65 km.

The SKA will be observing at frequencies covering 50 MHz to 14 GHz in the first two phases of its construction. When complete, it will have a total collecting area of a large fraction of one square kilometre and will be the world's largest radio telescope. SKA1-Low will operate in the redshift range $z \sim 6-28$ and will allow for high resolution (from scales of arc-minutes to degrees) direct imaging of the 21 cm line, to possibly even higher redshifts with SKA2-LOW (Koopmans et al., 2015). SKA will be trying to answer these five Key Science Programs: Probing the Dark Ages; Galaxy Evolution, Cosmology and Dark Energy; The Origin and Evolution of Cosmic Magnetism; Strong Field Tests of Gravity Using Pulsars and Black Holes; and The Cradle of Life.

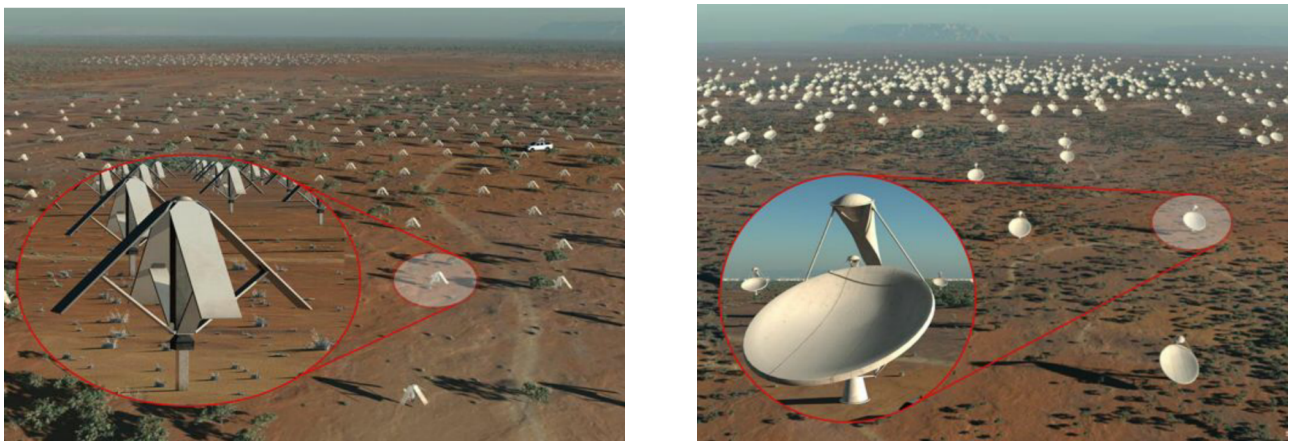


Figure 2.7: Artist's impressions of the two SKA receptors, the low frequency sparse aperture arrays to cover the frequency range from 50 to 350 MHz (Left) and the Offset Gregorian Antennas to cover the frequency range 350 MHz to 14 GHz (Right). Credit: Swinburne Astronomy Productions with enhancements by MAG

2.3 Calibration

Calibration has always been a keystone for astronomical observations, and with highly sensitive instruments, the characterization and development of precision calibration techniques for 21 cm cosmology has been expanded, leading to several calibration methods being proposed (Liu et al., 2010; Ewall-Wice et al., 2016; Dillon & Parsons, 2016; Barry et al., 2016; Ewall-Wice et al., 2017; Nikolic & Carilli, 2017; Dillon, 2017; Dillon et al., 2018; Grobler et al., 2018; Orosz et al., 2019; Kern et al., 2020; Dillon et al., 2020; Gorthi et al., 2021). Calibration methods for 21 cm observations have two main categories: 'Sky-based' calibration and 'redundant' calibration. Sky-based calibration involves fitting the measurements to the simulation of the sky and instrument models. However, since this method depends on the sky model, it suffers from contamination of the EoR power spectrum when calibrating on incomplete sky models (Ewall-Wice et al., 2016; Barry et al., 2016). Alternatively, redundant calibration has been introduced as a way to minimise this dependency on sky models because it uses the internal consistency of the array to fit these measurements during calibration (this is explained clearly in the next chapter).

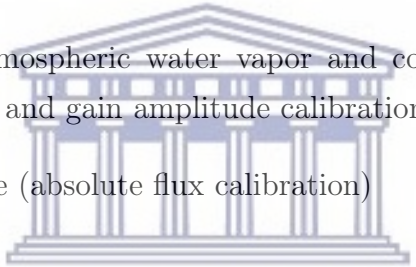
The number of radio telescopes built is rapidly increasing and they are becoming more ambitious. With an interferometer, it is not the case that the Fourier Transform of the observed visibilities will provide nice images. This may be due to

several factors: atmosphere (ionosphere, troposphere, water vapour); antenna/feed (system temperature, primary beam, pointing, position/location); digitiser/correlator (auto-levelling, baseline errors) and/or low noise amplifier plus conversion chain (clock, gain, phase, delay, frequency response).

Because of these effects, there arises a need to mitigate these effects by using calibration. In the calibration process, there is an effort to measure and remove the time-dependent and frequency-dependent atmospheric and instrumental variations. It uses these models to calculate corrections to the data. Many astronomical applications use the interferometer to measure the amount of interference (or coherence) in the incident radiation field to obtain information about the source morphology on angular scales (or spatial frequencies) sampled by the interferometer (Boden, 2007).

Most interferometers follow these general steps in calibrating their data:

- Correct frequency-dependent telescope response (bandpass calibration).
- Remove effects of atmospheric water vapor and correct time-varying phases/amplitudes (phase and gain amplitude calibration)
- Set absolute flux scale (absolute flux calibration)



However, these steps may differ depending on the type of array – how it is built, and which type of observations are made. Also, given the sophistication of the new instruments, the traditional calibration methods will need to be revised, leading to new calibration techniques for the new interferometers to reach their design sensitivities. Self-calibration (self_cal) (Nikolic & Carilli, 2017) or redundant calibration (redcal) (Dillon et al., 2020) have now become the proposed calibration methods to handle the sensitivity of these instruments because they allow the instrument to be calibrated off complicated sky emission structures, which is ideal (Liu et al., 2010). However, redundant calibration is still not without errors; 21 cm spectral contamination can be caused by (e.g.) systematic phase offsets in the calibration solutions because redundant calibration absorbs antenna position offsets into the calibration solutions (Joseph et al., 2018).

2.3.1 Redundant Baseline Calibration

The difficulty in observing the 21 cm line during the epoch of reionization, is due to the challenge the separating the cosmological signals from astrophysical foregrounds. This challenge has become a trigger for the construction of highly redundant low-frequency radio arrays (Liu et al., 2010; Deboer et al., 2017; Dillon et al., 2020). These configurations provide an alternative calibration strategy, "Redundant Baseline Calibration", and boost sensitivity on specific spatial scales. The Hydrogen Epoch of Reionization Array (HERA), shown in Figure 2.6, uses the internal consistency of the array to calibrate its data. So HERA is specially built with dishes that are arranged into a regular pattern so that pairs of receivers with the same position vector between them (same length and orientation) see exactly the same signal from the sky, which is called redundancy (Liu et al., 2010; Dillon & Parsons, 2016; Dillon et al., 2018; Grobler et al., 2018; Kern et al., 2020; Dillon et al., 2020).

Ultimately redundant baseline calibration is a process of finding a solution to a system of equations written in the form

$$V_{ij}^{obs}(\nu) = g_i(\nu)g_j^*(\nu)V_{ij}^{true}(\nu) + n_{ij}(\nu) \tag{2.7}$$

where $V_{ij}(\nu)$ is the visibility measured for the baseline between antennas i and j ($V_{ij}^{obs}(\nu)-$ is the observed and $V_{ij}^{true}(\nu)-$ is the expected visibilities), $gi(\nu)$ is the complex bandpass associated with antenna i and $n_{ij}(\nu)$ is the Gaussian-distributed thermal noise on that visibility. Accurate estimation of $g_i(\nu)$ is essential to 21 cm cosmology. By definition, the baselines of antennas i,j and k,l are considered to be perfectly redundant when their observed visibilities are equal, also taking into account their antenna gains, as shown in the equation

$$\frac{V_{ij}^{obs}}{g_i g_j^*} = \frac{V_{kl}^{obs}}{g_k g_l^*} \tag{2.8}$$

and not when $V_{ij} = V_{kl}$.

So the idea behind redundant baseline calibration is that, by building a highly redundant array like HERA, we are measuring the same modes on the sky, therefore the same baselines, over and over again. So this simply means this method is building

a big-linear system of equations, therefore the main goal is to minimise the χ^2.

$$\chi^2 \equiv \sum \frac{|V_{ij}^{obs} - g_i g_j^* V_{i-j}^{sol}|^2}{\sigma_{ij}^2} \tag{2.9}$$

Since it can model the unique visibilities(V_{i-j}^{sol}) and the gains(g_i and g_j^*), The notation $i-j$, as referred in Dillon et al. (2020), is intended to signify that any given correlation should depend only on the relative positions of the two antennas. The unique visibilities(V_{i-j}^{sol}) is a stand-in for V_{ij}^{true} shown in equation 2.7. In redundant baseline calibration, we try to solve for V_{i-j}^{sol} and g_i simultaneously, in the end making sure that it is as close to the real data as possible given the noise (σ_{ij}^2, noise variance). It is also important to note of all the parameters in this equation, V_{i-j}^{sol} is the only parameter fixed across redundant baselines, the only free parameters are the gain solutions for each antenna, g_i and g_j^*. However, since redundant baseline calibration does not explicitly need the sky model, after minimizing χ^2, there are some degeneracies that leave χ^2 unchanged (Zheng et al., 2014; Dillon et al., 2018, 2020). There are 4 of these degeneracies per polarisation and frequency that need to be solved when running redundant calibration.

- The overall amplitude ($g_i \rightarrow A g_i$, $V_{i-j}^{sol} \rightarrow A^{-2} V_{i-j}^{sol}$), then $g_i g_j^* V_{i-j}^{sol}$ is unchanged. If there is an error in the overall amplitude, the brightness of the sky will change, making the sky appear artificially bright.

- The overall Phase ($g_i \rightarrow e^{i\Psi} g_i$) the changes in g_i and g_j^* always cancel out. This degeneracy exists for both sky based calibration and redundant calibration, this is also because the χ_{sky}^2 (for the sky based calibration) responds the same way.

- The East-West tip-tilt ($g_i \rightarrow g_i e^{i\Phi_x x_i}$, $V_{i-j}^{sol} \rightarrow V_{i-j}^{sol} e^{-i\Phi_x \Delta x_{ij}}$), then $g_i g_j^* V_{i-j}^{sol}$ is unchanged for all baselines.

- The North-South tip-tilt ($g_i \rightarrow g_i e^{i\Phi_y y_i}$, $V_{i-j}^{sol} \rightarrow V_{i-j}^{sol} e^{-i\Phi_y \Delta y_{ij}}$), then $g_i g_j^* V_{i-j}^{sol}$ is unchanged for all baselines. The last last two degeneracies represent the phase gradient and errors in their parameters can make sources appear offset from their true positions due to a shift of the sky image.

The overall amplitude, the East-West tip-tilt and the North-South tip-tilt can be

solved by absolute calibration using a sky model as a reference. But since the overall phase is merely an arbitrary convention with no physical significance, it does not need the solved using the sky model (Dillon et al., 2020). The process of redundant calibration takes place in three steps, firstcal, logcal and lincal+abscal. Firstcal finds one delay and one phase offset per antenna and per polarization (but not per frequency), it uses the sky model to find a starting point for the rest of redundant-baseline calibration. Logcal (logarithmically linearized redundant-baseline calibration) finds an approximate per-frequency calibration solution and then finally refining that solution with lincal, which iteratively searches for the minimum value of χ^2 by first restricting degrees of freedom (Dillon et al., 2020). Lincal has been shown to produce less biased results compared to logcal (Liu et al., 2010).

2.4 Calibration and Foreground Cleaning

The ongoing and future experiments mentioned in chapter 2.2, which are primarily aimed at detecting the 21 cm signal, with their new level of precision, have led to the requirement of new calibration techniques for those interferometers to reach their design sensitivities. This is especially crucial given that the astrophysical foregrounds are around 4-5 orders of magnitude brighter than that of the cosmological 21cm signal (Morales & Hewitt, 2004; Santos et al., 2005; Furlanetto et al., 2006; de Oliveira-Costa et al., 2008). This foreground problem could be solved either: (a) by subtracting a model foreground components (made directly from the data or simulation based on the sky+instrument models) and using the resulting residual data to estimate the 21cm signal, known as foreground removal (Zaldarriaga et al., 2004; Wang et al., 2006). (b) by restricting the foregrounds within the wedge-shaped region in the ($k_\perp$, $k_\parallel$) plane leaving an 'EoR Window' with cosmological signal free of foreground bias, known as foreground avoidance (Datta et al., 2010).

Regardless of the approach to mitigate foregrounds, the issue is more complicated if the spectral structure imparted by the instrument cannot be calibrated-out or restricted to a limited part of Fourier space. It has made precision calibration recognised as a necessity to perform EoR Power Spectra observations. Several studies have been carried out, to tackle the problem of separating the cosmological signal from astrophysical foregrounds in 21 cm observations by proposing different calibration methods (e.g. (Mitchell et al., 2008; Liu et al., 2010; Dillon et al., 2020)).

Traditionally *self_cal* has been the go-to method for bandpass calibration. This is the iterative process of forward-modelling a source catalogue, solving for gains, imaging, and updating the source catalogue (Braun, 2013). This Self-calibration is done on sources whose Signal-to-Noise (S/N) ratio on each baseline is of the order of at least 5 or so. It is conceptually very similar to a basic calibration. The main difference is that the source model is generally more complex than simply assuming it is a point source at the phase center.

Ewall-Wice et al. (2016) and Barry et al. (2016) showed that even a small model error in the foreground can have a drastic effect on the instrumental calibration and can bias the 21 cm power spectrum. These modelling errors end up creating spectral structures on calibrated short baselines beyond what the instrument normally imparts because the intrinsic chromaticity of the long baselines ends up leaking into the gain solutions of the shorter baselines. Ewall-Wice et al. (2016) further states that their analysis of the chromaticity of longer baselines motivates possible solutions to noise modelling problems in sky-based calibration. Their proposed strategy is to down-weight the contribution of long baselines to the gain solutions that are applied to short baselines. While Barry et al. (2016) suggests limiting the number of degrees of freedom of calibration.

According to Byrne et al. (2019), which follows the work done by Barry et al. (2016), shows that errors caused by incomplete sky models introduce some errors during calibrations, even in the limit of perfect antenna positioning and identical beams. Byrne et al. (2019) also highlights how redundant calibration is affected by sky models errors and how they are actually worse for redundant arrays. This is because it follows a two-step process; 'relative' calibration and absolute calibration. The former solves for the antenna gains, and the latter constrains the degenerate parameters described in 2.3.1, which were left unsolved by 'relative' calibration. However, redundant calibration still suffers from its fundamental assumption that redundant baselines measure the same sky visibility, as this is not true in the case where, the primary beam responses are different (Choudhuri et al., 2021), and there are antenna-to-antenna variations in dish construction and placement. Orosz et al. (2019) proposed a redundant-baseline calibration strategy that relies predominantly on short baselines to mitigate these antenna-to-antenna variations.

For our project, we propose a modified redundant-baseline calibration strategy that uses a machine-learning clustering algorithm to cluster baselines within a re-

dundant baseline group based on their visibility solutions instead of relying solely on the 'perfect' redundancy of the array. It means that we are trying to mitigate for any instrumental effect that could be present during observations by taking into account not only the redundant design of the array but also the visibility solutions that are solved for during calibration. So by splitting a redundant baseline group into sub-groups using the clustering algorithm, we are effectively increasing the number of degrees of freedom to be solved for in Equation 2.9. Therefore, this method includes all the steps involved in redundant calibration with the only difference being the redundant baseline groups used for calibration, as shown in Figure 4.2. This work is also motivated by the fact that for non-moving drift scan telescopes like HERA, characterization of the primary beam is especially difficult because standard beam-calibration routines are not applicable (Cornwell et al.). Therefore this means that we need a way to mitigate for whatever effects (beam non-redundancies) might affect this telescope by improving the calibration method.

3 Machine Learning

Machine learning (ML) is a branch of artificial intelligence (AI) and computer science that uses data and algorithms to learn and make predictions based on that dataset. In the digital age, where almost every one of us has a digital footprint, thereby leading to a lot of data to sift through, ML has become the most promising scientific method to handle this data. ML techniques have become an acceptable methodological approach in many fields like Astronomy (Baron, 2019), banking (Huang et al., 2020), Medicine (Litjens et al., 2017) and many other fields. The integration of ML techniques has become a necessity, especially in the field of Astronomy, where we are entering the big-data era due to the high-end observatories being built. Bringing in large amounts of data and their complexities, like LSST (now renamed the Vera C. Rubin Observatory) and eventually the Square Kilometre Array (SKA).

In machine learning, there are two main approaches, supervised and unsupervised learning. The main distinction between these two approaches is the use of labelled datasets. Supervised learning deals with labelled data, that is, data where we already know what the output should be. Therefore, the goal of supervised learning is to learn from a given sampled dataset and desired outputs (labels) and then give the best approximation of the relationship between input and output observ-

able in the data (Murphy, 2012; Jordan & Mitchell, 2015; Goodfellow et al., 2016b). On the other hand, unsupervised learning deals with unlabeled data, that is, we do not have any prior knowledge of what the output should be. The main goal of unsupervised learning is to infer the relationship between the data points, if there is any, within a given dataset (Jordan & Mitchell, 2015).

3.1 Supervised vs Unsupervised Learning

Supervised learning is done in the context of classifying, where we try to assign labels to outputs from input examples, or regression, where we try and predict a continuous value from input features. Since supervised learning is also known as a predictive model, it can be thought of as trying to fit in a function. For example, in giving the algorithm a dataset with a set of labels, we are basically building a function that can best describe the input data. By providing the algorithm with new inputs, we are trying to find the best outputs that closely fit that function. Now, this leads to one of the most drawbacks of supervised learning, to fit such a function, the model is highly dependent on what is deemed "correct" outputs.

It is also important to note that this "correct" output is entirely dependent on the training set (given data with its corresponding labels), however, incorrect acquisition of the dataset and wrong/noisy labelling will clearly reduce the effectiveness of the model (Medar et al., 2017). There are a lot of supervised learning approaches including but not limited to Decision trees (Quinlan & Cameron-Jones, 1993; Cardie, 1993), K-nearest neighbors (Aha et al., 1991), Logistic regression (Le Cessie & Van Houwelingen, 1992) and Neural Networks-deep learning (LeCun et al., 2015; Goodfellow et al., 2016a).

In Unsupervised Learning, the model is not trained at all. A dataset is given to the algorithm for the computer to learn the patterns in data. Unsupervised learning is done in the context of clustering, that is, grouping data points based on their similar traits or features. This method is frequently used to learn new aspects of the data that have not been observed yet, often leading to new discoveries (Goodfellow et al., 2016b). Most of its application is in anomaly detection (Chandola et al., 2009), which is a branch of ML where a data point or points do not conform to a well-defined normal behaviour of a given dataset. It is often applied to banks to

detect fraudulent transactions against their clients, and in astronomy, it is applied to detect new astronomical objects like transients, supernovas and anything that looks out of place on the night sky(Mahmood et al., 2010; Xiong et al., 2011).

Unsupervised learning techniques include K-means clustering, Hierarchical clustering, Principle component analysis (PCA) and Apriori Algorithm. The most popular being K-means clustering, which segments data points within a dataset into clusters/sub-groups based on features. Some algorithms like PCA are generally applied to reduce the dimensionality of a given dataset. PCA is often applied to discern the importance of particular features used for analysis. By trimming down the features, we are thereby increasing the interpretability of the dataset but at the same time minimising information lost.

3.2 K-means clustering

K-means clustering is one of the most popular unsupervised machine learning algorithms, first introduced by MacQueen et al. (1967) and Lloyd (1982) as a technique for pulse-code modulation. In recent times the algorithm is usually used in pattern recognition and data mining (e.g.(Huang, 1998; Singh et al., 2011; Wu, 2012; Windarto, 2017; Hossain et al., 2019)). K-means is used for clustering unlabelled data into k clusters, and it does this by trying to separate samples in n groups of equal variance. It is a distance-based algorithm that uses N data points defined in space of dimension I, to find the minimum distance between them and the nearest cluster centroids/centers, to split them into K clusters. This distance is calculated using the Euclidean distance in an I-dimensional space defined as,

$$\begin{aligned} d(p,q) &= \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+\ldots+(q_I-p_I)^2} \\ d(p,q) &= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2} \end{aligned} \tag{3.1}$$

Each cluster is parameterized by a vector $m^{(k)}$ called its mean. We will represent

the data points as $\{x^{(n)}\}$ where subscript n runs from 1 to N, with each x having I components x_i. We have a metric that defines distances between points by assuming a real space that x lives in, for example,

$$d(x, y) = \frac{1}{2}\sum_i (x_i - y_i)^2 \tag{3.2}$$

Lets consider a two-dimensional dataset that we are trying to split into 2 clusters (i.e. K=2). The K-means algorithm follows a 4 step process to cluster the data:

1. **Initialization**: Set K means $\{m^{(k)}\}$ to random values that is, place n points into the feature space to represent the initial cluster focal point. These points represent initial cluster centroids.

2. **Assignment step**: Each data point n is assigned to the closest cluster centroid. We denote our guess for the cluster $k(n)$ that the point $x(n)$ belongs to by $\hat{k}(n)$.

$$\hat{k}(n) = \underset{k}{argmin}\{d(m(k), x(n))\}. \tag{3.3}$$

An alternative, equivalent representation of this assignment of points to clusters is given by 'responsibilities', which are indicator variables $r_k^{(n)}$. In the assignment step, we set $r_k^{(n)}$ to one if mean k is the closest mean to datapoint $x^{(n)}$; otherwise $r_k^{(n)}$ is zero.

$$r_k^{(n)} = \begin{cases} 1 & \text{if } \hat{k}^{(n)} = \text{k} \\ 0 & \text{if } \hat{k}^{(n)} \neq \text{k} \end{cases} \tag{3.4}$$

In the unlikely event where two means are exactly the same distance from a data point, $\hat{k}^{(n)}$ is set to the smallest of the winning $\{k\}$.

3. **Update step**: When all points have been assigned, recalculate the positions of the K centroids by adjusting them to match the sample means of the data points that they are responsible for.

$$m^{(k)} = \frac{\sum_n r_k^{(n)} X^{(n)}}{R^{(k)}} \tag{3.5}$$

where $R^{(k)}$ is the total responsibility of mean k

$$R^{(k)} = \sum_n r_k^{(n)} \tag{3.6}$$

If $R^{(k)} = 0$, – that is, if we find means with no responsibilities, then we leave the mean $m^{(k)}$ where it is

4. **Repeat the last two steps**: Until the assignments of the K centroids does not change or until a specified number iterations is reached.

This 4 step process is what makes K-means easier to understand, thereby allowing more room to modify and improve the method, evidenced by the large number of publications in the last 50 years or so. However even with its simplicity K-means still has its problems especially when it comes to dealing with steps 1 and 4, whether that is on two or multidimensional data (which has its own host of effects).

3.2.1 Issues with K-means clustering

Despite how popular K-means, with its variety of applications it still has its drawbacks, which have been extensively studied in literature, with some suggesting ways to mitigate them. The main drawbacks for K-means is that:

- It assumes you already have a deep knowledge of your data, therefore that you know the number of clusters your data should be split into. But when dealing with real world data we often have little knowledge of the data, which is why K-means is often applied as an exploratory analysis method (Jain et al., 1999).
- It is an iterative process, which makes it very sensitive to the initial starting point, which can lead to different final results every time the method is run.
- It converges finitely at a local minimum. This means that it chooses the correct option that is in the neighbourhood instead of looking at the bigger picture (choosing the most optimal solutions that fits perfectly).

The fact that the value of K has to be supplied by the user is one of the main drawbacks that plague not only K-means but also other clustering algorithms. This

drawback could either be a major or a minor problem since this is highly dependent on the user's knowledge of the data. If you are already very familiar with the data, especially when you already know the number of classes, then this should not be a problem, so will now only have to deal with the other problems. However, in real-world situations, that knowledge is not readily available, which makes it difficult to ascertain the value of K. Nevertheless, several methods have been used to try to gauge the number of clusters: the most common methods are, the Elbow method, the Silhouette method, to the most complicated; the Akaike's information criterion; Bayesian inference criterion and Cross-validation. A review of these methods of determining the K in K means was done by Kodinariya & Makwana (2013). Other extensive studies have been carried out to try and validate the number of clusters by doing a comparative study of cluster validity indices (Arbelaitz et al., 2013).

The elbow method is visual and it is the most commonly used method for finding K in K-means, although it is a little naive in its approach. It calculates the Within-Cluster-Sum of Squared Errors (WSS) for different values of K usually starting at $K = 2$ and chooses the K for which WSS becomes first starts to diminish. Then a plot is made with the K-values on the x-axis and the WSS on the y-axis. The K-value corresponding to the point where the plot begins to diminish (forming an elbow) is the optimal value for K for that data set. The major problem with this method is that it leaves a lot to interpretation because sometimes there is no elbow at all or this "elbow" cannot always be unambiguously identified.

Another popular method for determining K is the Silhouette Method (Rousseeuw, 1987), which is another graphical approach, however, unlike the elbow method, it shows a concise graphical representation of how well each object has been classified and has shown success in (Pollard & Van Der Laan, 2002; Arbelaitz et al., 2013). This silhouette shows which objects lie well within their cluster, and which ones are merely somewhere in between clusters. It does this by calculating the distance of a point between its cluster and another cluster, using any distance metric, such as Euclidean distance or the Manhattan distance. This measure is called the silhouette value, ranging from -1 to 1. If more points have values closer to 1 means that they belong to that cluster, if it is closer to 0 then it could belong to either cluster and if the values are closer to -1 then it does not belong to that cluster or the clustering configuration may have too many or too few clusters. This gives a better view of the natural number of clusters present in a data set, also allowing the silhouette to

be maximized by re-scaling the data using feature weights that are cluster-specific (De Amorim & Hennig, 2015).

Even with brief descriptions of these methods it is obvious that how they go about determining the optimal K-value, the final clustering will depend on the initial cluster centres chosen.

Since K-means is an iterative process, the end product is always determined by the initial steps. Since the introduction of K-means, several initialisation methods have been proposed, including choosing the first K data points, randomly choosing K data points in the dataset and dividing the data points randomly into K clusters (Random partitions). The first two methods were proposed by MacQueen et al. (1967), with the first one suffering from the ordering of the data. The second method seems to be more sensible, since choosing the centroids at random and is likely to pick a good candidate as cluster centers. However, this does not prevent it from picking any points that are close to each other or any outliers or even offer any mechanism to mitigate it (Anderberg, 1973), which will in turn affect the convergence of K-means. In recent years, there has been some introduction to more advanced techniques like density-based initialization and Intelligent initialization.

Given that the different approaches to initializing K-means will eventually yield different results, a comparative study of efficient initialization methods for the k-means clustering algorithm has been done by Celebi et al. (2013). They tested eight of the commonly used initialization methods: Forgy's method (Forgy, 1965), MacQueen's second method (MacQueen et al., 1967), maximin (Gonzalez, 1985), Bradley and Fayyad's method with J = 10 (Bradley & Fayyad, 1998) , k-means++ (Arthur & Vassilvitskii, 2006), greedy k-means++ , Var-Part , and PCA-Part (Su & Dy, 2007). They tested these methods on a large and diverse collection of data sets using various performance criteria, which then found that the Bradley and Fayyad's method was consistently the best performing, even on smaller data sets; $N < 10000$ (with greedy k-means++). By default, the K-means clustering algorithm on scikit-learn uses k-means++ as its initialization method.

The other problem faced when implementing this algorithm is there is no way to make sure that the points are clustered correctly, which is one of the biggest disadvantages of dealing with unsupervised learning. This is because k-means converges on a local (find references in Bottou & Bengio (1995)) instead of a global minimum.

It means that for the 4th step, even if the assignments of the K centroids do not change, it does not mean that those are the correct cluster centroids. The concept of local and global minimum can be clearly explained using figure 3.1, which shows that K-means tends to choose the local minimum as the ideal model, even though the optimal model might be present or has not been reached yet, given the number of iterations/how quickly K-means converges. Given that the convergence of k-means at a global minimum is not guaranteed, it is, however, insured at a local minimum (Selim & Ismail, 1984).

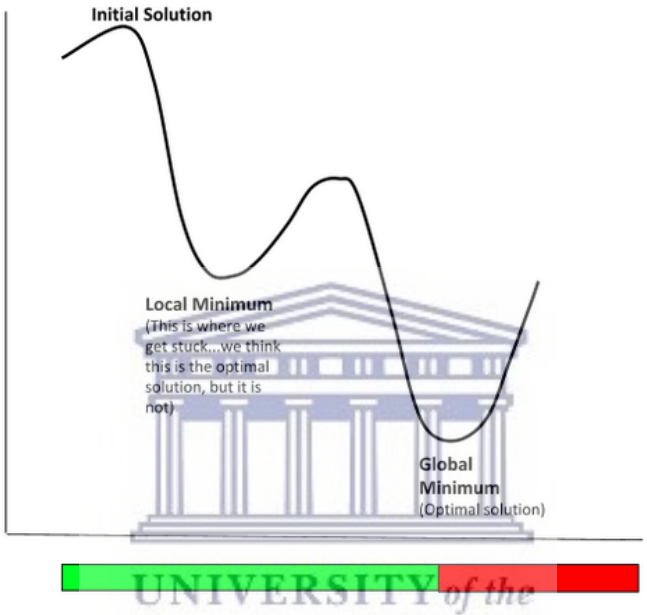


Figure 3.1: A plot showing the local and global minimum for different K-means models (automaticaddison, 2019). This diagram is an analogy of how the K-means clustering algorithm chooses what is in the neighbourhood instead of focusing on the larger picture.

3.3 Hierarchical clustering

Hierarchical clustering, also referred to as also called hierarchical cluster analysis or HCA, is another clustering technique, used to split similar objects of a data set into groups called clusters. As the name suggests, it does this by building a hierarchy of clusters, following two types of approaches: Agglomerative (a "bottom-up" approach) by starting each observation in its cluster, we have pairs of clusters that are merged (subsequently updating the intercluster distances) as we move up the hierarchy. And Divisive (a "top-down" approach), unlike the previous method

we start all observations in one cluster, and then we split the cluster recursively as one moves down the hierarchy. This method is highly visually representative with its results represented using a dendrogram, a diagram that shows the hierarchical relationship between objects shown in figure 3.2.

Several hierarchical clustering techniques have been proposed, and some literature reviews that include hierarchical clustering have been published (Olson, 1995; Xu & Wunsch, 2005; Murtagh & Contreras, 2012). Since all these methods are either agglomerative or divisive, the decision to merge or split the objects and a measure of dissimilarity between sets of observations is necessary. In a lot of these methods, an appropriate metric is required to measure this dissimilarity (a measure of distance between pairs of observations). These metrics include the: Euclidean distance, Maximum distance, Squared Euclidean distance, Manhattan distance and Mahalanobis distance. Another parameter needed is the linkage criterion which specifies the dissimilarity of sets as a function of the pairwise distances of observations in the sets. Both these parameters have a large impact on the overall performance of the clustering algorithm. For example consider two dimensional data points, $a = (0,\ 0)$; $b = (0.5,\ 0.5)$ and $c = (0,\ 1)$, as we compare two different distance metrics the Euclidean distance (equation 3.1) and the Manhattan distance metric defined as,

$$d(p,q) = \sum_{i=1}^{n} \mid q_i - p_i \mid \tag{3.7}$$

from this we can see that using the Manhattan metric, $d(a,b) = d(a,c)$, however when using the Euclidean metric $d(a,b) \neq d(a,c)$, and consequently the Euclidean distance $d(a,c)$ is shorter than the Manhattan distance $d(a,c)$. It shows that the choice of an appropriate metric will influence the shape of the clusters, especially when some elements are relatively closer to one another under one metric than another.

Of the two hierarchical clustering methods, the Divisive Hierarchical Clustering (DHC) technique is rarely used in the real world, so the dendrogram shown in figure 3.2 (right) is showing how elements are clustered in Agglomerative Hierarchical Clustering (AHC). In the plot, we have six observations, and according to the definition of AHC, we start from the bottom-up, assuming that each observation is its unique cluster. In the second step, a distance metric is used to calculate the

distance between the points, preferably using a distance matrix for the i-th and j-th elements, recording the results in the i-th row j-th column. The shortest distances between these elements are merged to form a new cluster, thereby merging columns and rows, and the distances are updated. Now the way these distances are updated is different, and it's also one of the most important parameters when running AHC, called linkage criterion.

A simple AHC adopts the single-linkage clustering. As an example we have clusters $\{A\}$, $\{B\}$, $\{C\}$, $\{D\}$, $\{E\}$ and $\{F\}$ we can see that $\{E\}$ and $\{F\}$ are closer to each other, so at first we end up with these clusters $\{A\}$, $\{B\}$, $\{C\}$, $\{D\}$ and $\{E, F\}$. The minimum distance in the cluster $\{E, F\}$ is now used as an updated position for cluster $\{E, F\}$. To merge further we need to take the distance between $\{D\}$ and $\{E, F\}$, and therefore define the distance between two clusters. Usually, the distance between two clusters P and Q is one of the following:

- Single-linkage clustering - The minimum distance between elements of each cluster: $min\{d(x, y) : x \in P, y \in Q\}$
- Complete-linkage clustering - The maximum distance between elements of each cluster:

  $max\{d(x, y) : x \in P, y \in Q\}$
- Average linkage clustering - The mean distance between elements of each cluster:

  $$\frac{1}{|A| \cdot |B|} \sum_{x \in P} \sum_{y \in Q} d(x, y)$$

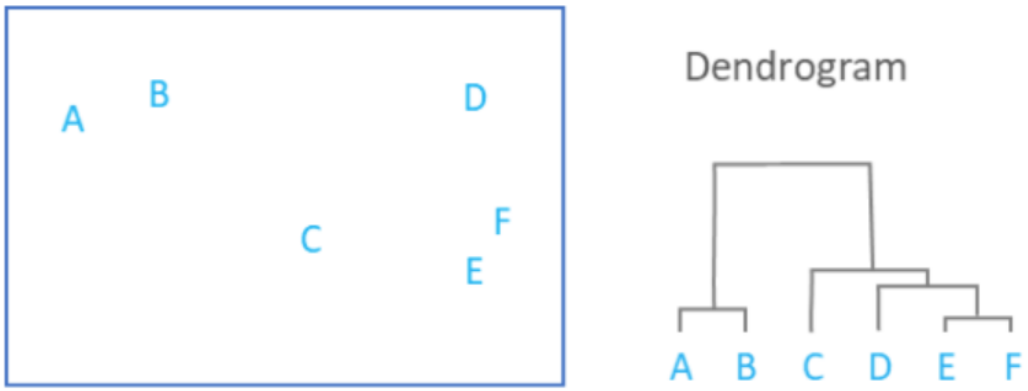


Figure 3.2: The plots shows the hierarchical clustering (right) of six observations shown on the scatter-plot to the left. The heights reflect the distance between the clusters (Bock, 2020).

In the case where the minimum distances are equal, a pair is randomly chosen, leading to structurally different dendrograms being generated. Alternatively, all tied pairs may be joined at the same time, generating a unique dendrogram (Fernández & Gómez, 2008).

Because of how AHC does its clustering, especially with its dependence on the two input parameters (the distance metric and the linkage criterion), it requires very little knowledge of the data set from the user (Murtagh, 1984), making it very useful for exploratory analysis. This also leads to its most important advantage of not requiring the number of clusters we needed to split the data. That being said, dendrograms cannot definitively tell you the number of clusters, however, in some cases, like the given example, the dendrogram can suggest the number of clusters.

4 Methodology

This chapter explains how the data used in this study was created and also explains the steps taken to improve the redundant baseline calibration code (redcal). This chapter is organised as follows. In subsection 4.1 we describe the standard model of the array and different types of non-redundancy perturbations considered. In subsection 4.2, we give a brief description of the machine learning clustering algorithms we are using, and how the data is structured. In subsection 4.3 we describe how the redcal code was modified, including different options considered in the modification.

4.1 Simulation

The simulations were run using a non-redundant pipeline, by Dr Phil Bull and Samir Choudhuri found on GitHub (https://github.com/philbull/non-redundant-pipeline). This pipeline is for simulating, calibrating, and analysing data from non-redundant arrays, based on the HERA stack. Most of the simulation parameters are taken from Choudhuri et al. (2021), and they explain the simulation and its dependencies in detail. Therefore, this chapter will give a brief overview of the simulation.

Most of the data presented in this work is for simulations run with a bandwidth of 100 - 120 MHz with 120 frequency channels and 10 time samples. Because we used very few time samples, means we observed the sky for a few minutes, covering the LST range of 9.1 - 9.3 hours. The sources that we observed have a flux falling in the range of about 5 - 15811 Jy. These sources are neither brightest nor the faintest in the sky, making them ideal for our simulation. The choice of LST range and observed sources was done to reduce the simulating time because bright sources dominate most of the emissions used. The drawback with this small observing time is that in that time tot all of these sources will rise above the horizon within the simulated observing time.

4.1.1 Redundant array layout

For our data, we simulate an array like HERA, and like HERA it has a nominal array centre of $30°43'17"S$ and $21°25'42"E$, operating as a drift scan instrument, pointing at zenith. So the simulated HERA instrument has closed packed arrays

with a large number of redundant baselines to improve the instrument sensitivity on the relatively large angular scales needed for 21 cm experiments. So unlike HERA, which will comprise of 350, 14-m dishes/antennas when complete, with the shortest baseline of 14.6 m. For our simulation, we emulate the hexagonal shape of HERA but with different antenna numbers. We have simulations with 10, 24, 75 and 124 antennas, each serving a particular purpose in our analysis process, which are detailed in section 5.1. All receivers are assumed to be in the same plane and equidistant from each other, with no significant elevation changes or position errors between them.

The choices for the different array layouts were made specifically for the validation of the clustering algorithm and the modified calibration code (logi_cal), especially for the 10 and 24 antenna arrays. Which is a reasonable number of antennas to provide several well-populated redundant baseline groups with a few different lengths and orientations. It also limits the computational time for a simulation run 10 time-samples with 10 frequency channels, using the 10 and 24 arrays, and that gives a total of 45 and 276 baselines, respectively. The small array size allows us to better explore the data, making sure get what we expect from the simulation, as well as testing the visibility data on a machine learning algorithm. Once validations are carried out on the small array, we then move to a much bigger array with 75 and 124 dishes. These simulations might be computationally expensive, however, it is much better than the alternative of simulation the entire HERA array. So these simulations on the large array give a fair representation of how the real HERA array would react when exposed to the different cases of non-redundancy. This will also give some measure of the relative importance of the array size concerning the clustering algorithm used for calibration.

4.1.2 Models of primary beam non-redundancy

In this subsection, we explain how we went about introducing non-redundancy to a perfectly redundant system in a form of primary beam perturbations. To do this, we used a pre-existing code on GitHub (https://github.com/philbull/non-redundant-pipeline), which already has the configuration files needed to add those perturbations. Each of these yaml files represents the type and level of perturbation added to a perfectly redundant system. To model the antenna-to-antenna variation in the primary beam, we add some perturbations to the true (model) visibility for antenna

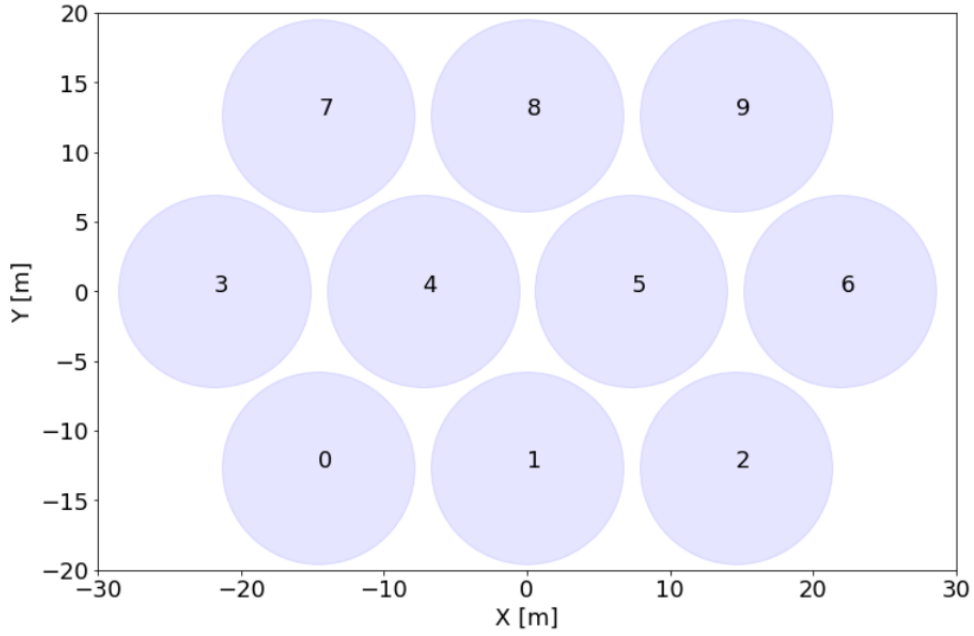


Figure 4.1: This figure shows the array used in our simulation with 10 antennas arranged in a hexagonal shape. Each antenna has a diameter of 14m and separated by 14.6m.

pairs (i, j) in the equation

$$V_{ij}^{true}(\nu) = \int_{\Omega} B_{ij}(\theta,\ \nu) I(\theta,\ \nu) e^{-2\pi i \boldsymbol{u}_{ij}\cdot\theta} d^2\Omega \ , \tag{4.1}$$

where B_{ij} is the primary beam power pattern, $I(\theta,\ \nu)$ is the specific intensity in the dimensional sky plane position θ and frequency ν, and $\boldsymbol{u}_{ij}$ is the baseline vector for antenna pairs (i, j). The simulation pipeline has many case parameters to consider, but 4.1 shows cases that are prioritised, particularly case3a, since most testing and validations for this study were done using this case of non-redundancy:

	Beam perturbation type	**Degree of perturbation**
Case 1	Sidelobe	$\sigma_{SL} = 0.05$
Case 3	Stretched beam	(a) Gaussian, $\sigma_m = 0.01$
Case 4	(a) Ellipticity	Ellipticity ($e = 1\%,\ 2\%$)
	(b) Ellipticity + Rotation	Ellipticity ($e = 1\%,\ 2\%$) and Rotation $\alpha \sim Unif[0°, 360°]$
Case 5	Stretched baseline length	bl_len = $+10cm$

Table 4.1: The table shows a summary of the models of primary beam non-redundancy used in our for this project. The table is a recreation of table 1 in Choudhuri et al. (2021), with modifications to only include perturbations in the study.

Case1: Sidelobe-only perturbations— This simulation only includes the side-lobe perturbation effect of the antenna-to-antenna variations without modifying the main-lobe at all. To modulate the fiducial beam pattern beyond a zenith angle θ_{ML}, we use a low-order Fourier series ($N = 8$) with randomly-chosen coefficients, where θ_{ML} defines the 'edge' of the main-lobe. For this model, we assume that the side-lobe variations can be quite complex, potentially shifting the location and depth of nulls in the beam. The beam perturbation, is defined as

$$\tilde{b}(\theta, \phi) = b(\theta, \phi) \left(1 + c_{SL}\sigma_{SL}\Theta(\theta) \sum a_m sin(2\pi m\theta/L)\right), \tag{4.2}$$

where $L = \pi/2$ is the period which corresponds to the angle between the zenith and the horizon. The modulation is normalised by $c_{SL} = [max(y) - min(y)]^{-1}$ where the summation term y in eq. 4.2 is rescaled to span [-1, +1] independent of any chosen values of the coefficient $\{a_m\}$. For the project we consider a case where the amplitude $\sigma_{SL} = 0.05$

Case3a: Stretching the primary beam— In this simulation, we are stretching the whole primary beam, but in the same amount in the x-y direction, changing the overall angular size of the beams from antenna to antenna, using Gaussian random numbers. So this simulation changes the entire beam for the configuration files on GitHub, perturbation is added (set to 'True') for this simulation, but the perturbation scale is set to zero (meaning we do not want to perturb the side lobes). So for different ways of perturbing the primary beam, for each antenna:

(a). m is drawn from a Gaussian distribution with mean unity and standard deviation $\sigma_m = 0.01$ or 0.02.

(b). m is drawn from a Uniform distribution between [−0.02, +0.02] (which is roughly comparable in width to the $\sigma_m = 0.01$ case above).

Case4: Ellipticity and rotation of the primary beam — Considering the axisymmetric nature of the basic beam model, this case allows for perturbations in ellipticity and rotation to model beam squint and feed rotation effects by simply remapping the coordinates of the axisymmetric beam.

Case5: Stretching the baseline length— This is the only simulation with positional errors. The perturbation includes the stretching of the baseline length up to 10 cm. So in this case, the beam patterns are unmodified, the only thing modified is the antenna positions to get that 10 cm perturbation.

4.2 Classifications

For classifying the levels of non-redundancy in the HERA array, we use a machine learning clustering algorithm called k_means clustering discussed in subsection 3.2, and Agglomerative Hierarchical Clustering (AHC) discussed in subsection 3.3. In k-means, the clustering is done on a set P on n points, but for our case, we want to cluster a set of visibility profiles for a given baseline, meaning we want to cluster lines. In (Marom & Feldman, 2019) they discuss how to cluster a set of L of n lines, in a sense of visibility per frequency channel. Here the distance from a line to a center c is the closest Euclidean distance to c over all the points on the line. Since we want to cluster these visibility profiles (lines), we will treat the points that make up those profiles as dimensions, to use during clustering.

The k-means clustering algorithm is a package taken from scikit-learn (a machine learning software library for the Python programming language) and is solved using either Lloyd's (Lloyd, 1982) or Elkan's algorithm. The k-means algorithm only requires two inputs: (1) The data we want to cluster and (2) The number of subgroups we want to segment a particular group into. As a default, the algorithm uses k-means++ as an initialisation method, meaning for a chosen k-value, the k-centroids will be chosen at random every time the code runs. This is also coupled with the fact that k-means always converges to a local minimum (see Bottou & Bengio (1995)), implying that the results are not reproducible, simply meaning that the makeup of resulting clusters will change with every iteration.

To cluster our baselines from a redundant baseline group into even more redundant subgroups, we use K-means clustering. As inputs, we use the absolute value of the visibility for each baseline. For example, for the second simulation, we have a set of visibilities for each frequency channel. For a simulation that has only one time sample we have point sets $t_1 = \{v_1, v_2, ..., v_n\}$, where v is the visibility for frequency channel number 1 to n. Now K-means will cluster the baselines based on their similarities between the points in the set t1, for each baseline in a redundant baseline

group. However for multiple time samples, in the case of the first simulation we have $T = [t_1, t_2, ..., t_{10}]$, where each t has 10 values corresponding each frequency channel. Now the input for k-means will be a T, where instead of stacking the visibility solutions corresponding to each channel, we just append the time samples into one 1D array i.e. in this simulation, each baseline will have 10×10 visibility solutions. To summarise, the input data is the absolute value of visibilities as a function of time and frequency.

If the data has more than two variables (two dimensional) like our data set, say it has three variables, x,y and z. The distance between these points and the centroids are calculated in this way: $d = |x_2 - x_1| + |y_2 - y_1| + |z_2 - z_1|$

NOTE: I am going to refer to the number of sub-groups we are clustering the RBGs as k or k-values, this also include when we are talking about the number of sub-groups when using AHC.

4.3 Modifying RedCal

This subsection will discuss how we went about trying to improve the redundant calibration (redcal) code. Going forward for context and clarity, refer to the Hera_cal documentation on GitHub(HERA_team). For our modification, we focused mainly on two functions; (1) *get_reds*, which gives us a list of lists (sorted by baseline length) of redundant baseline tuples. (2) *redcal_iteration*, which performs redundant calibration (performs a single iteration of the redundant calibration algorithm, which must be repeated many times to converge to a solution).

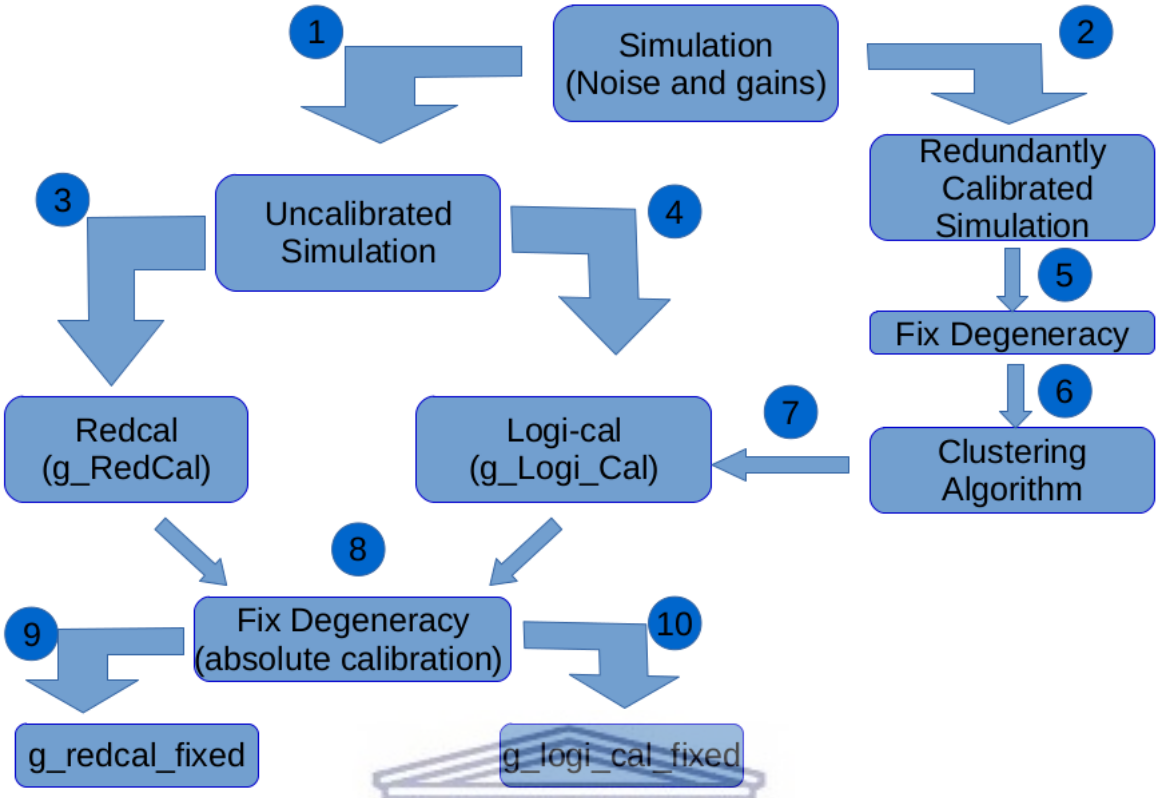


Figure 4.2: This flow chart represents a summary of how the data is calibrated using redcal and logi_cal. This figure illustrates the extra steps (namely 2, 5 and 6) that logi_cal has to go through to calibrate the data compared to redcal. It is also important to note that this figure does not show the internal processes that are involved in running redundant calibration.

In modifying *get_reds* (into a new function called *custom_get_reds*), we include K-means which will cluster baselines within a redundant baseline group into even more redundant sub-groups. Now the *custom_get_reds* will still give us that list of lists as the original *get_reds*, however, it will now include lists with even more redundant baseline groups. The visibility solutions we get from running redcal the first time are used as inputs to cluster those baselines within a redundant baseline group, as shown in step 5 of the flow chart. Then the resulting lists from *custom_get_reds* as inputs, in place of the line that calls the *get_reds* functions, on the *redcal_iteration* function.

4.4 Ways of clustering Redundant Baseline Groups (RBGs)

In clustering the redundant baseline groups (RBGs), we implement different methods to investigate a variety of questions. The main issue we face when dealing with a clustering algorithm is that we do not know the number of subgroups we are supposed to have for each of the redundant baseline groups. The baselines in a RBG have specific visibilities, meaning each RBG will respond to k in k-means in different ways. Because of this predicament, we suggest 4 different ways of dealing with this problem (considering k-means).

4.4.1 Option 1: Cluster RBGs using the same k-value

In this method, all the specified RBGs are clustered using the same k-value. This means that for a HERA75 simulation which has about 228 RBGs if we specify that we want to cluster the first 60 RBGs, each of those groups will be clustered into the same number of subgroups i.e.the same k-value. This is a very simplistic approach since it assumes that all the RBGs respond relatively the same to the same k-value, without any regard to the visibilities present for each baseline in those RBGs. This method assumes that there is an optimal k-value that fits all the scenarios for a particular case of non-redundancy.

4.4.2 Option 2: Cluster RBGs using different k-values (Uniform)

This method involves clustering all the specified RBGs using different k-values. Redundant baseline calibrations usually uses RBGs that are ordered based on the lengths of the baselines present in those groups. Because of that the first RBGs (for example in that HERA75 simulation), usually have a higher number of baselines in their groups and that number decreases as you go to RBGs at a higher index. For example, the 1st RBG has 112 baselines, and the 60th RGB has 52 baselines. Because of this, it is only logical to think that the first few RBGs should have more clusters and the later ones should have fewer clusters. So as an example, for the 60 specified RBGs, the groups could be split into four, 0-15 will use k=6, 15-30 will use k=5, 30-45 will use k=4, and 45-60 will use k=3. It is also important to mention that the shorter baselines are split more times, and the longer baselines that split fewer times. For this method, we try to see if groups RBGs respond differently to

different k-values and how it affects the calibration solutions.

4.4.3 Option 3: Cluster RBGs using different k-values (Random-k)

This method involves clustering all the specified RBGs using different k-values, but unlike the previous method, the clustering of the RBGs are completely random. So each of the first 60 specified RBGs are using a random k-value ranging from 2 to 6. This method is, however, not that practical and reliable especially given a higher range of values. It is mainly because there is a higher margin of error between iterations of the same calibration for the same parameters. This could however be reduced if we limit the range of k-values to be considered in the calibration, to only two choices in k-values (either 2/3, 3/4, 4/5, 5/6 or any combination of two k values). For this method, we try to see how individual RBGs respond to different k-values and how it affects the calibration solutions.

4.4.4 Option 4: Cluster RBGs using the same k-values (Random-baseline label)

This method is very different from the other three methods because this does not include a clustering algorithm. In this method, we use the same k-value, but since there is no clustering algorithm, the baselines are clustered at random. Each baseline within a redundant baseline group is given a random label (within a set range), and then the baselines with the same label are considered to be in the same cluster. This method is mainly introduced to try and validate the need for a clustering algorithm. This is to observe if the is any improvement in the calibration method, or whether it is due to the clustering algorithm or the simple action of just splitting the RBG. So if the results we get from this method are at least similar to the ones from the previous three approaches, then it shows that there was no need for a clustering algorithm to begin with.

5 Data Analysis

This chapter's main focus is how we went about analysing the data from the multitude of simulations we ran. The chapter is organised as follows. In section 5.1 we try to visualise the data so we could visually observe and understand what redundancy is, and to make sure that the simulation outputs what we expect. In section 5.2 we describe how the clustering algorithm works on our data set. In section 5.3 we describe different summary_statistics (input) we use for the clustering algorithm, that best describe the data for a given simulation. In section 5.4 we describe the best clustering algorithm that best fits our data set. And finally In section 5.5 we compare the results we get when we calibrate the data using redcal and logi_cal.

5.1 Data Visualisation

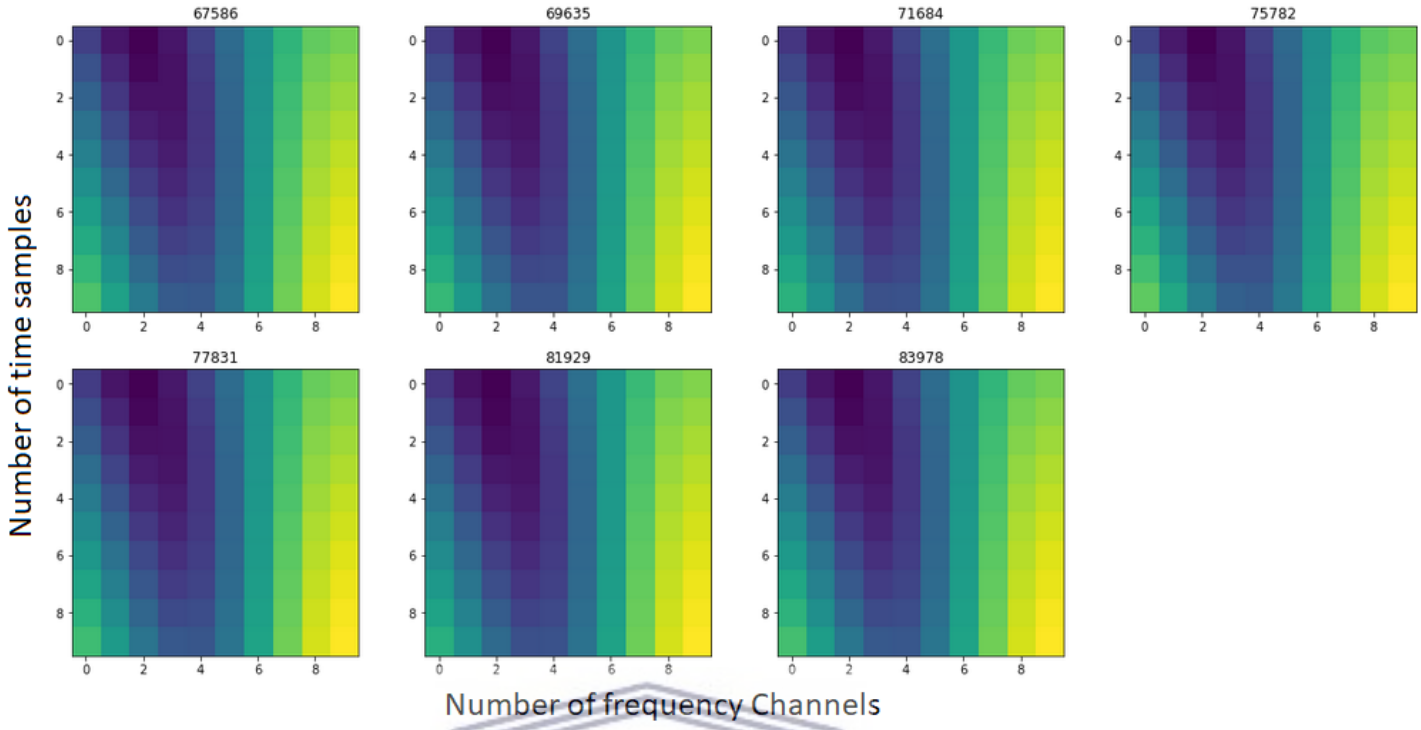


Figure 5.1: This is the 2D visibility or waterfall plot (number of frequency channels vs time samples) of baselines within a Redundant Baseline Group, with each number on the plot title representing the baseline ID (to help identify antenna pairs making up the baseline, 67586=(0,1), 69635=(1,2), 71684=(3,4), 75782=(4,5), 77831=(5,6), 81929=(7,8) and 83978=(8,9)). These baseline IDs are just unique numbers that help in identifying antenna pairs. These baselines look similar to one another, clearly showing that they are redundant.

Figure 5.1 is for a simulation that was run with nfreq=10 and ntimes=10, using 10 antennas i.e. 45 baseline/correlated measurements, which resulted into 20 Redundant Baseline Groups (RBG). These RBGs are in an array arranged by length, from short to long-baseline length. The results below are for a redundant baseline group with baselines of length 14.6m, this group has the most baselines (with 7), excluding the group with auto-correlated baselines (with 10). We exclude the auto-correlated baselines because they have correlated noise terms, and thus are likely to be much noisier than other correlated measurements (Liu et al., 2010).

The fact that the visibility of these baselines looks almost indistinguishable from each other through visual inspection, meant we had to find a mathematical way

to distinguish between them i.e. find the level of non-redundancy. In Figure 5.2 we show the mean and the standard deviation for the RBG described above and both their statistical values can help us in quantifying the non-redundancy of these baselines. From this plot, we can see that these baselines within this RBG have a high level of redundancy, given the range of values from the standard deviation plot, since it shows a little variation in the visibility solution. Another way to confirm the non-redundancy for this case is to find the typical percentage level of non-redundancy for this group. We calculated this using the equation $std/mean$, which is (size of fluctuations) / (size of signal), resulting in roughly 3.0% of non-redundancy in this group.

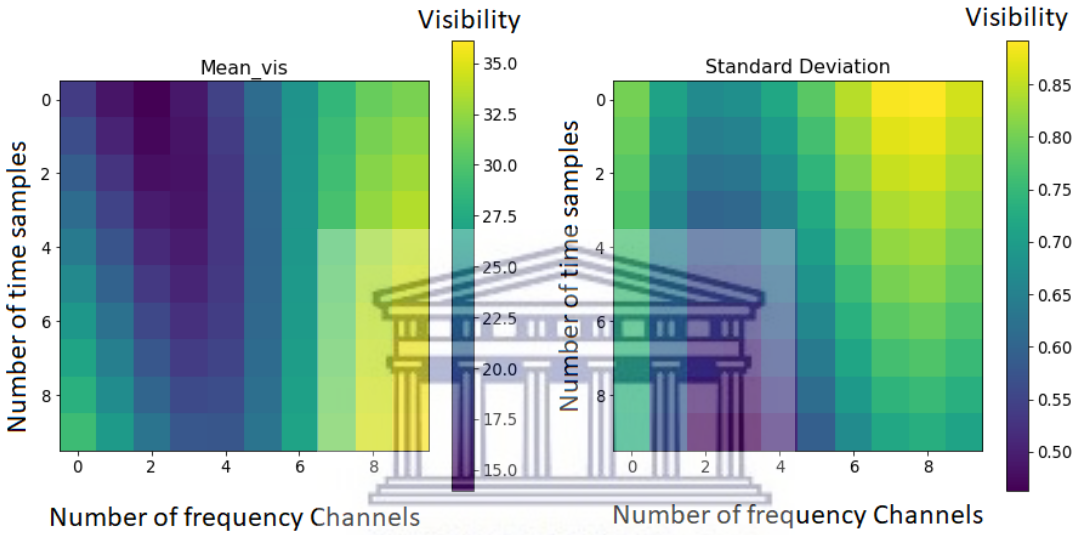


Figure 5.2: This is a waterfall plot of the mean and standard deviation of the Redundant Baseline Group for all time samples and all frequency channels used. The color-bar shows the visibility solution.

The small simulation above is presented to show what redundancy looks like and how calibrated HERA is to have this many baselines with very similar visibilities. However, HERA will be operating with more baselines, frequency channels and time samples. For Figures 5.3 and 5.4 we show results for a simulation with the same number of antennas (10), but with more frequency channels and time samples, to test the effect they have of the level of non-redundancy. This simulation was run with nfreq=120 and ntime=600, meaning the simulations covers the LST range 9.2 - 15.8 hours. For the same parameters we ran this simulation for two cases, Case1: has added noise and antenna gains and Case2: has no added noise and antenna

gains. Case1 is what we could expect when making an observation with HERA since we know that during astronomical observations there is always noise, either internal or external. Case2 act as a control, this is what we would see if there were no internal or external factors affecting the data.

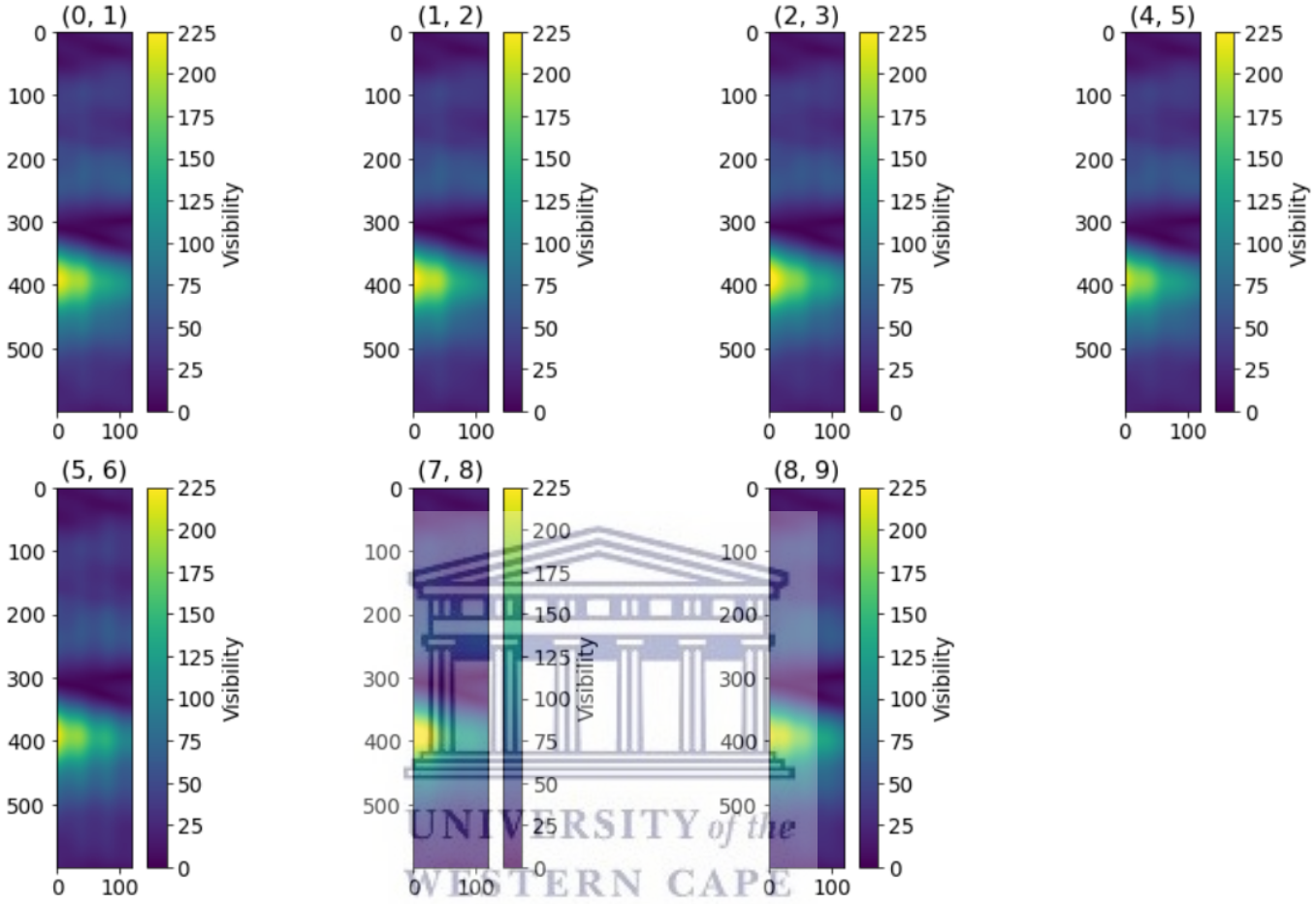
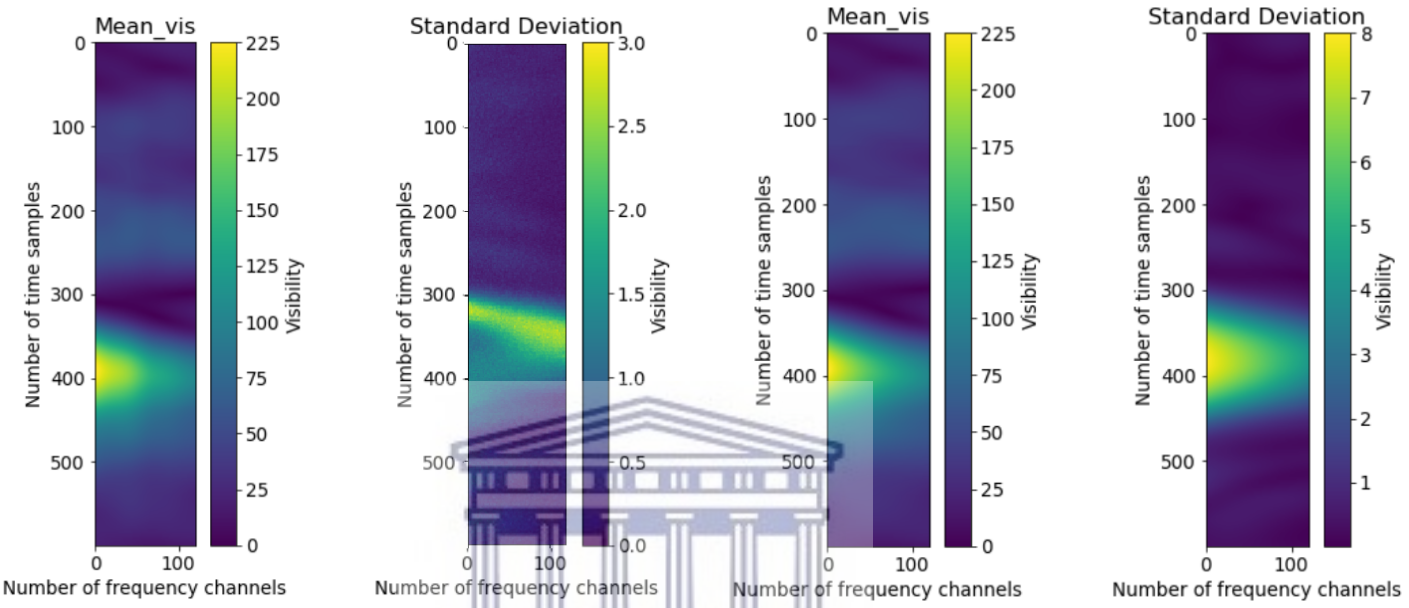


Figure 5.3: This is the 2D visibility or waterfall (number of frequency channels vs number of time samples) plot of baselines is the Redundant Baseline Group, with each number of the plot title representing the antenna pairs making the baseline. These baselines look similar to one another, clearly showing that they are redundant. The x-axis is the number of frequency channels and the y-axis is the number of time samples.

Figures 5.4(a) and 5.4(b) show the waterfall plot of the mean and standard deviation of the visibility solutions for case1 and case2. Case1 is a simulation with added noise and gains, and case2 is a simulation without added noise and gains. using the mean and std values from Figures 5.4(a) and 5.4(b), we find that case1 has a level of non-redundancy at 4.117% and for case2 is 9.994%. It is important to note that case1 is calibrated for noise and gains using redcal. One thing that is

peculiar about the percentages of non-redundancy for both cases in Figure 5.4, is that we get the reverse of what we expected. For example, we expected the noisy data (case1) to have a higher percentage of non-redundancy and higher standard deviation compared to the data for case2. But this discrepancy might be due to redcal, which tries to find gains that are more redundant than they appear to be, which might result in the noisy data hiding some of its non-redundancy since noise makes things look more equal.



(a) Case1: Added noise and gains **(b)** Case2: No added noise and gains

Figure 5.4: This is a waterfall plot of the mean and standard deviation for both cases. The color-bar shows the visibility solution. (a) Show an instance where the simulation has added noise and antenna gains (observed visibility). (b) Shows a case where the simulation has no added noise and gains (true visibility).

5.2 Classification (Testing clustering algorithm)

In this section we investigated if we could cluster the baselines within a redundant baseline group into sub-groups using the visibility solutions. In short we are checking the validity of using visibility solutions for clustering the baselines.

For the first part of our classification, we used the k-means clustering algorithm, and as a test set, we used a 2D array, which constitutes a single time sample (containing visibility values per frequency channel) for each baseline. By 2D-array, we

mean for each time sample (ntimes), there is n number of frequency channels (nfreq), then if nfreq=120, we would have 120 dimensions per baseline that K-means has to deal with. Since K-means is an unsupervised machine learning algorithm, for us to evaluate its performance, we have to cluster the baselines ourselves.

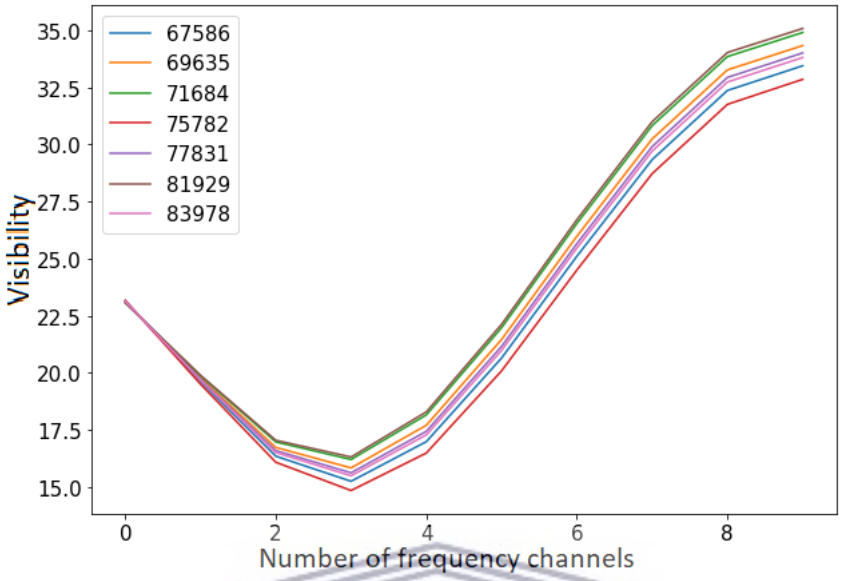
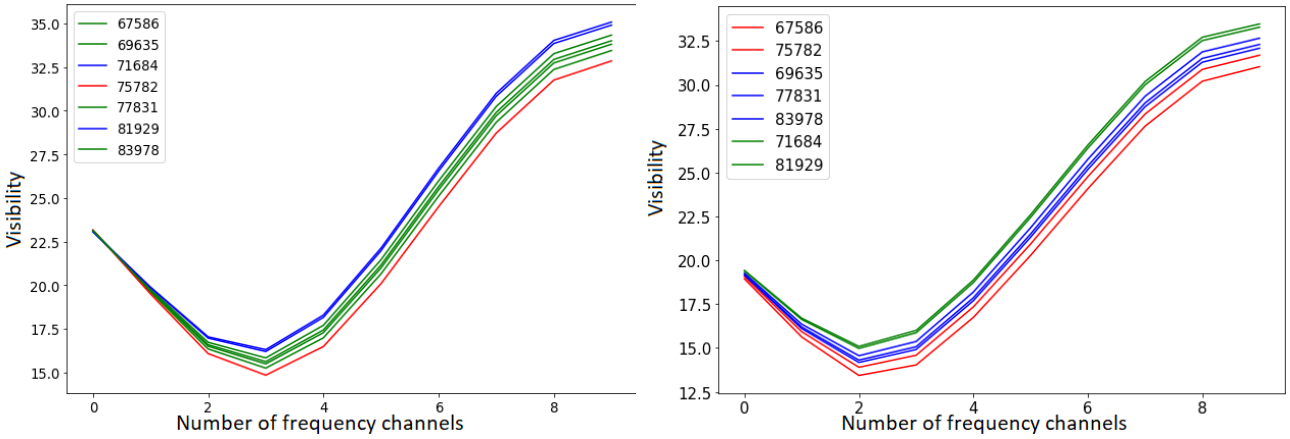


Figure 5.5: This is a single time sample of the each baseline in the RBG as function of frequency, with each number on the legends representing the baseline ID. As we can see there is a level of non-redundancy between these baselines.

Figure 5.5 shows that even if there is some level of non-redundancy between these baselines, there are still some almost redundant baselines. Because of this, we can try to classify baselines that are similar into groups. Based on what we see in Figure 5.5, we use three groups to cluster the baselines. The plot in Figure 5.6(a) was done to see what we would expect when using the clustering algorithm in classifying the baselines. The plots show that the clustering algorithm works since our classifications look very similar to the grouping done by the clustering algorithm, with only one not matching our manual classification.

Now that we know that the clustering algorithm works, we expanded our array from 10 antennas (small array) to 24 antennas (big array). For the big array, the same method as the small array was followed, which resulted in a 1.38% level of non-redundancy, from a redundant baseline group of 19 baselines (Figure 5.7). In classifying the baselines, unlike the small array, the baselines of the big array were clustered into four groups. But for the big array, it is hard to manually cluster these

baselines by just looking at the profiles on the plot (Figure 5.7), so this means for arrays with more antennas, we have to depend on the clustering algorithm. However, even if we have to depend on the clustering algorithm, Figure 5.5 has shown that k-means has the ability to cluster the baselines based on their visibility solutions, given the 6/7 score we get when comparing clustering in 5.5(a) and 5.5(b).



(a) Manually

(b) K-means clustering

Figure 5.6: A plot showing how similar baselines are grouped together. Plot (a) showing how we would group these baselines by color-coding them manually to specify that they belong to the same group. Plot (b) shows how the clustering algorithm is classifying these baselines given the same data.

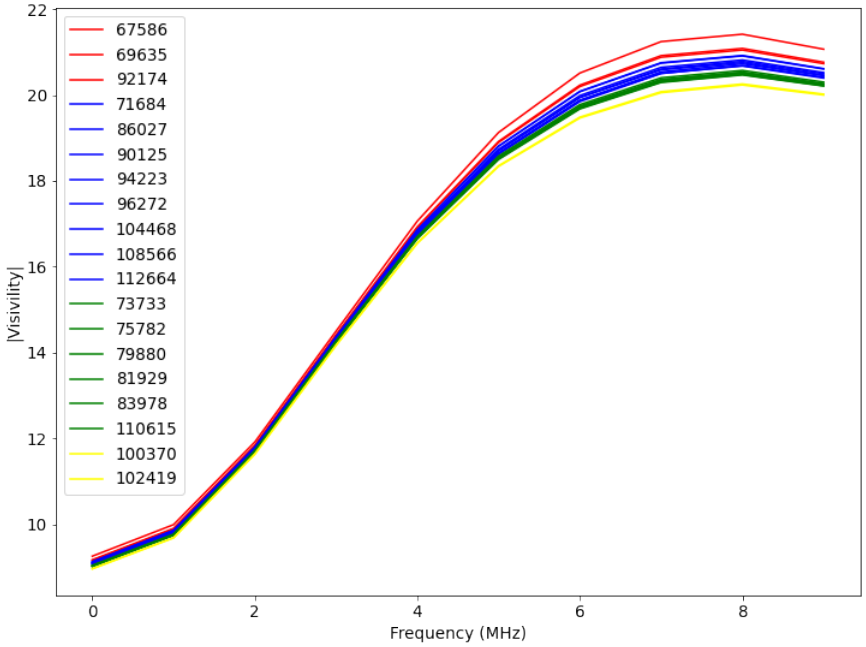


Figure 5.7: A plot of classified baselines by the K-means clustering algorithm from the big array (24 antennas) with 19 baselines in its RBG. The baselines are clustered into four clusters.

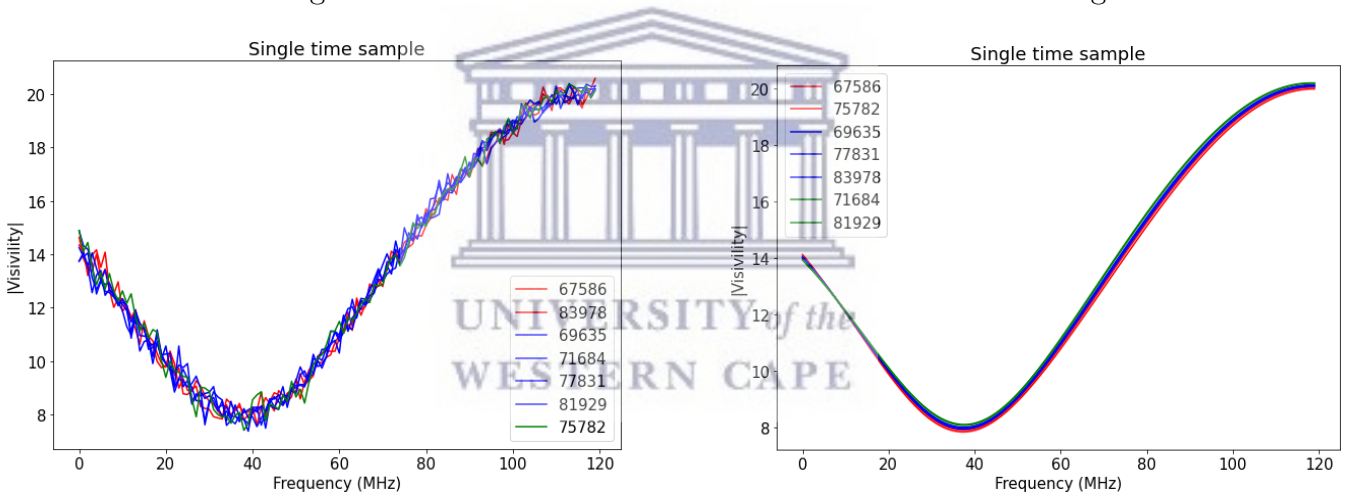
5.3 Comparing summary statistics

In this section, we are investigating two things, (1) the performance of the clustering algorithm on data with added noise and antenna gain, and (2) how we handle the input data for the clustering algorithm. The data used in this section is for Case3a (Stretching the primary beam) non-redundancy, this is for a 10 antenna array with 10 time samples (ntimes) and 120 frequency channels (nfreq). For the first point, we are checking if the clustering algorithm can deal with data that has added noise and antenna gains by comparing the clustering we get when we cluster baselines that have added noise and gains (case1) in them and ones that do not have noise and gains (case2). So the idea is if the clustering for case-1 matches the clustering for case-2, then it means that the clustering algorithm can handle noisy data.

For the second point, we have compared different summary statistics used in clustering the baselines. The summary statistics are how we choose to represent

the data, and these statistics were used as input for the clustering algorithms. We evaluate the success of these summary statistics by checking if the classification for Case-1 matches our own for Case-2. Using the same simulated data as the one used in Figure 5.8 to 5.10, we have summarized how each summary statistic behaves given the same simulated data and using the same clustering algorithm.

Figure 5.8 shows the results for the clustering of case-1 and case-2 data when using a single time sample as input. The idea of these plots is to make sure that the clustered groups for case-1 match that of case-2, since case-2 represent the "true" visibility solutions. That would mean that even if there is added noise and gains, the clustering algorithm can still cluster the baselines like there were no added noise and gains. From this plot, we can see that when the is added noise and gains, k-means struggles to produce high accuracy results on the clustering, since it manages to cluster 4/7 baselines in their correct groups. When we compare the groups for case-1 and case-2, baseline 83978 is labelled as red when it belongs to the blue group, baseline 75782 is green when it is red and baseline 81929 is blue when it's green.



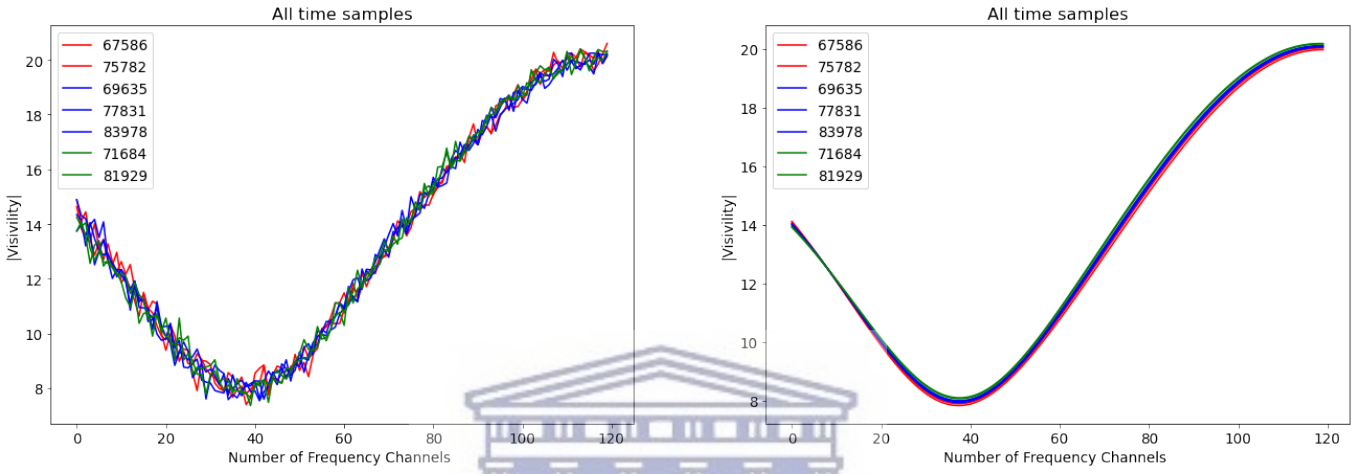
(a) Case1: Added noise and gains

(b) Case2: No added noise and gains

Figure 5.8: A clustering algorithm classification where the two cases are compared by plotting a single time sample of the number of frequency channels against the absolute value of the visibility.

Figure 5.9 shows the results for the clustering of case-1 and case-2 data when using all-time samples as input. The data has 10-time samples and was run with 120 frequency channels, so using all-time samples means we are appending the time samples into one array per baseline. So in terms of input for the clustering algorithm, it means we end up having 1200 data points (dimensions) per baselines. The plots

seen in Figure 5.9 are for 1-time sample, but the input used to cluster these baselines, is for when we used all-time samples as input. What we see in this plot is that even when there is added noise and gains, k-means can still correctly cluster these baselines, as we are seeing 100% accuracy. By that, we mean the clustered baselines for case-1 match that of case-2. Since the baseline in the red, blue and green groups contain the same baselines when comparing both cases. This result also means having more dimensions for the k-means improves the performance of this clustering algorithm.



(a) Case1: Added noise and gains (b) Case2: No added noise and gains

Figure 5.9: A clustering algorithm classification where the two cases are compared by plotting all time samples of the number of frequency channels against the absolute value of the visibility.

As we have seen with Figures 5.8 and 5.9 it hard to manually cluster the baselines especially when more frequency channels are used. It is especially impossible to make out any relationship between the visibility profiles of these baselines, seen in Figure 5.9 (a), that can enable us to confidently cluster these baselines. So one way to eliminate this problem is to use the RMS values of the visibility solutions of each baseline. Figure 5.10 shows the RMS values (y-axis) for each baseline, where the x-axis is just the index the baselines are in a list. This plot shows that it is now much easier to see similar traits between the baselines that we could not see with Figures 5.8 and 5.9. With this new found clarity, the baselines were clustered using RMS ranges, so any baseline in the range $> 5.18 = green'$, $5.18 < RMS < 5.12 = blue$ and $< 5.12 = red$. The top plots in Figure 5.10, show that when using these RMS ranges, the clustering does not match for case-1 (left frame) and case-2 (right frame).

The middle and bottom frames in Figure 5.10 are not using RMS ranges to cluster these baselines, but the cluster we see are using the labels we get from Figures 5.8 and 5.9. To put it simply, we are using k-means to cluster these baselines, and based on the labels assigned to them, we paint that picture on the RMS plot. We did this to check if these RMS ranges can be used as a reliable summary statistic. Since we have established that using all time samples as input produces the best result, we represent the labels we get from the clustering in the RMS plot at the bottom of Figure 5.10. It shows that there is no clear relationship between the baselines that are in the same cluster when looking at the case-1 plot. The results for all these summary statistics are recorded in Table 5.1, and what this shows is that whatever summary statistic is used, the clustered groups for case-2 match. It means the clusters we get are probably the correct clusters, and whatever statistic we use that can match that clustering for case-1 is the one we will use going forward.

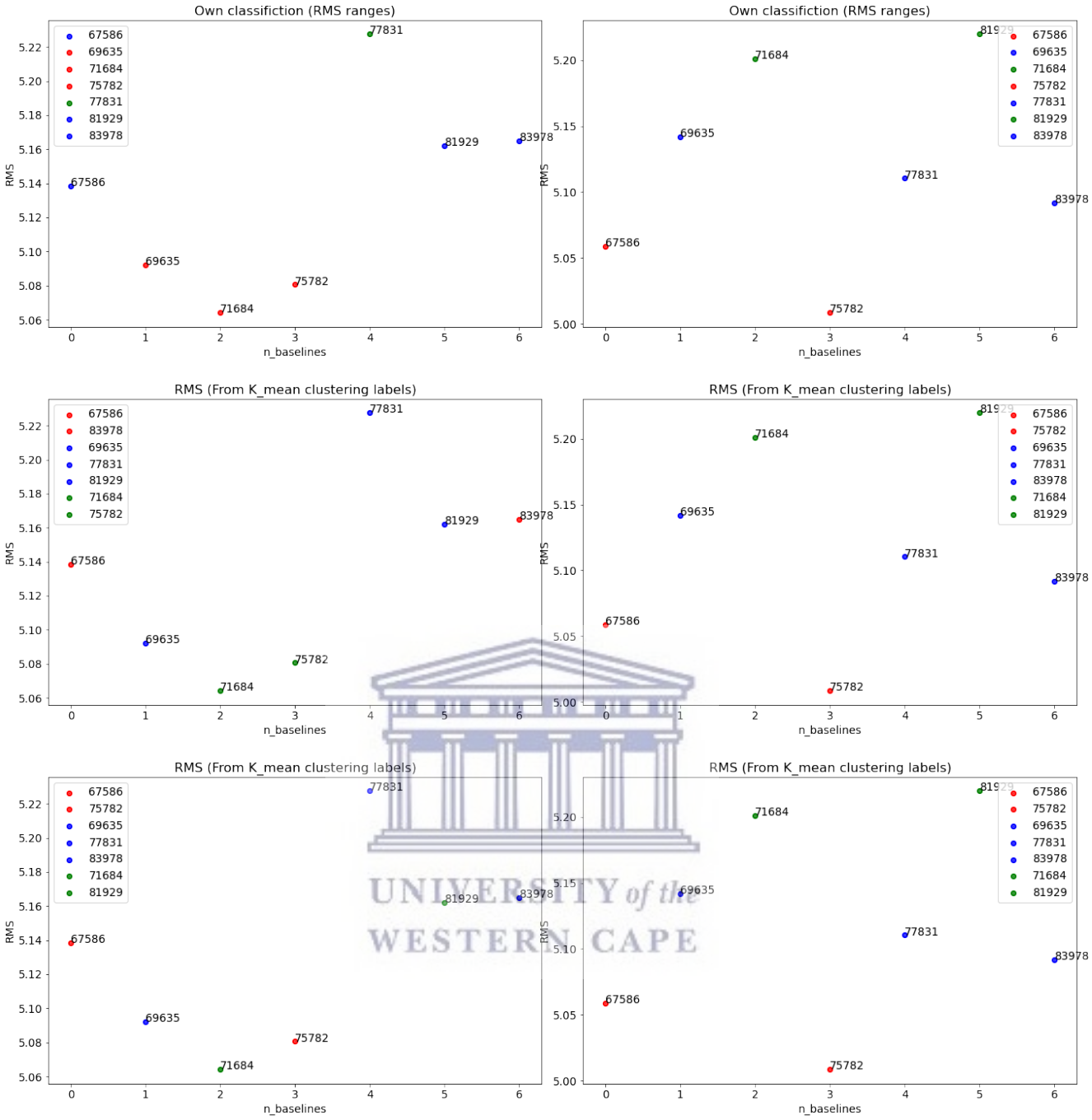


Figure 5.10: The RMS plots for the redundant baselines. The x-axis just represents the index of the baseline in a list, so its value is not significant. The frames on the left show case1 and the right frames show plots for case2. From top to bottom we have: baselines we classified on our own using RMS ranges; baselines classified from the labels of the k-means cluster algorithms for a single time sample and baselines classified from the labels of the k-means cluster algorithms for all time samples respectively.

Methods of clustering (Summary statistics)							
Own classification		Single time sample		ATS (appended)		ATS (Average)	
Case-1	Case-2	Case-1	Case-2	Case-1	Case-2	Case-1	Case-2
(0, 1)	(0, 1)	(0, 1)	(0, 1)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
(4, 5)	(4, 5)	(4, 5)	(4, 5)	(4, 5)	(4, 5)	(4, 5)	(4, 5)
(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)
(5, 6)	(5, 6)	(5, 6)	(5, 6)	(5, 6)	(5, 6)	(5, 6)	(5, 6)
(8, 9)	(8, 9)	(8, 9)	(8, 9)	(8, 9)	(8, 9)	(8, 9)	(8, 9)
(3, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)
(7, 8)	(7, 8)	(7, 8)	(7, 8)	(7, 8)	(7, 8)	(7, 8)	(7, 8)

Table 5.1: The table shows the list of summary statistics used. The numbers in the parenthesis represent the antenna pairs, making up the baselines within 1 redundant baseline group. Each of those antenna pairs is coloured depending on the group they belong to according to the clustering algorithm, and the type of summary statistic used for clustering. So for each summary statistic, if the colours match for between case-1 and case-2 then we consider that method to be the most optimal one to choose in our future clustering.

In the table we are trying to cluster the baselines in such a way that the groups for case1 match groups for case2 (True case). All time samples (3rd column from the right) is the best performing method, since its clustering matches for both case1 and case2 and also with my own classification for case2. But the term 'best performing' is subjective since we are dealing with an unsupervised machine learning algorithm, in a sense that for us to say the algorithm and the summary statistic performs well, we have to classify those baselines into groups ourselves in what we deem to be a 'good' classification and see if the algorithm gives the same results as us.

In this section we have shown that stacking time samples(and averaging over them per frequency channel) or just using only one time samples does not yield as good of results as when appending all the time samples into a 1D array. This is why for most of our simulation we use all the time samples as input on the clustering algorithm. This may be due to the fact that K-means has more data points to deal with when using all time samples instead of just one.

5.4 Comparing clustering algorithms

In this section we compare how different clustering algorithms cluster (groups) the baselines and we use the best summary statistic we discussed in section 5.3 as our input. We also compare how each algorithm interact and clusters different models of primary beam non-redundancy shown in Table 4.1.

Not only do we compare these algorithms, we also try to find the optimal number of clusters for each algorithm that will give the best accuracy. To do this comparison we use a confusion matrix, which is the most efficient way of dealing with the clustering of such a big array. A confusion matrix is a simple visualization tool for evaluating the performance of a classification system. It is in the form of a grid with one axis representing the correct labels (assumed, as discussed in the section 5.3) and the other axis representing the labels predicted by the classifier. Each entry in this grid is an integer value representing the number of times each actual class has been assigned to each predicted class. This grid can be displayed as an image that quickly visually represents the accuracy of the classifier. An ideal classifier has only diagonal values, meaning that all points are correctly classified. It is important to note that in defining the confusion matrix, we are using words like "classifier" and "classification", normally these words are reserved for explaining a supervised machine learning algorithm. In our case the correct wording should be "clustering", but in this section we are trying to validate the effectiveness of the clustering algorithm on working on noisy data. Therefore because of that we interchange the terms "clustering" and "classifying".

To get the accuracy for each clustering algorithm from the confusion matrix we use this simple formula:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.1}$$

for a binary classification, where True Positive (TP), True Negative (TN), False Positive (FP) and False Positive (FP). But for a Multi-Class Classification Confusion Matrix the accuracy can get a little tricky, but for a general accuracy of the entire

algorithm we use this simple equation:

$$\begin{aligned} Accuracy &= \frac{Sum\ of\ the\ true\ positive(diagonal)}{Total\ number\ of\ the\ sample} \\ Accuracy &= \frac{\sum_{i=1}^{n} TP}{n} \end{aligned} \tag{5.2}$$

The idea of doing a confusion matrix on an unsupervised machine learning clustering algorithm is a little bit unorthodox since our data is not labelled and the algorithm is not trained. So for us to get the confusion matrix, we must first find the 'true' labels, and for that, we used the labels we get when we cluster the data for case2 (No noise/gains). As we have shown in section 5.3 that case2 is an ideal case where our manual classification matches that of the clustering algorithm. Therefore given that we have shown that using all time samples as our summary statistic gives a 100% accuracy for the redundant baseline groups of 7 and 19 redundant baselines, it is only logical to assume that we might get the same results for large scales (more antennas, time samples and frequency channels).

It simply means that we do not need to manually cluster the baselines ourselves since it will be more challenging to do that on more baselines and also try to avoid imposing our prejudice and biases on the final clustering. **So the biggest assumption we have made going forward is that the 'true' labels are always correct even in a larger array**.

5.4.1 Accuracy, precision and recall

Considering what we want to achieve in clustering the baselines into even more redundant baseline groups, we have to examine what we care about in those clusters. Since this is a simulation we care about the accuracy, precision and recall, however when dealing with cases we do not know the true values precision and recall takes precedence. Even if this is the case, it is still difficult to decide which is better (high precision and low recall or vice-versa). Recall is just the ability of a model at detecting the positives or simply finding all the related cases within a data set.

So when we look at it, in clustering the baseline it is okay if a few baselines are

clustered in groups they do not belong into (false positive), but what if in doing so the baseline would have had a great impact on the improvement of the calibration solutions, when clustered into its designated group (true positive). Having baselines into groups they do not belong to (acting as outliers) has some drastic effects on the calibration solution, however since we are clustering within a redundant baseline group its effect is mitigated.

So we our main objective is to find out of all the positive predicted, what percentage is truly positive. In other words, out of all the baselines clustered in one group, what percentage of the baselines truly belong in that group i.e. the precision given by,

$$Precision = \frac{TP}{TP + FP} \tag{5.3}$$

Data:

We use data for a simulation that was run 60 times (ntimes) with 120 frequency channels (nfreq), with an additional parameter, $hex\_spec = [5, 12]$, where hex_spec represents the number of antennas to use in the hexagonal pattern of HERA, 5 is the number of antennas on the top and bottom part of the hexagon and 12 is the number of antennas in the middle part of the hexagon. This results in 124 antennas arranged in a hexagonal pattern, giving us a total of 7750 baselines, 124 of those being self-correlated. At the moment we considered a redundant-baseline-group with more redundant baselines, and that is the group with the shortest baseline length of $14.6m$ containing 109 redundant baselines.

5.4.2 K_means Clustering

K_means Clustering is the most popular machine learning clustering algorithm and is the one we used most in our classifications. Since we were implementing the algorithm on a smaller array we now implement the algorithm on a bigger array and use confusion matrix to do the evaluation of its performance.

Case 3a: Stretching the primary beam by 1%

Using more groups to cluster these baselines would have been ideal since it would

make the simultaneous calculations of the gains more accurate. We have tested the accuracy of the algorithm on the ideal number of clusters(groups) ranging from k=3 to k=9. The problem with $k \geq 4$ is not only about the low accuracy but the fact that its accuracy and the matrix would change every time we run the clustering algorithm. Therefore as the matrix shows, we opted to use k=3 since it was the most stable and has the highest accuracy (94%) compared to the other cluster numbers. The classification report shown below summarises the performance of the algorithm:

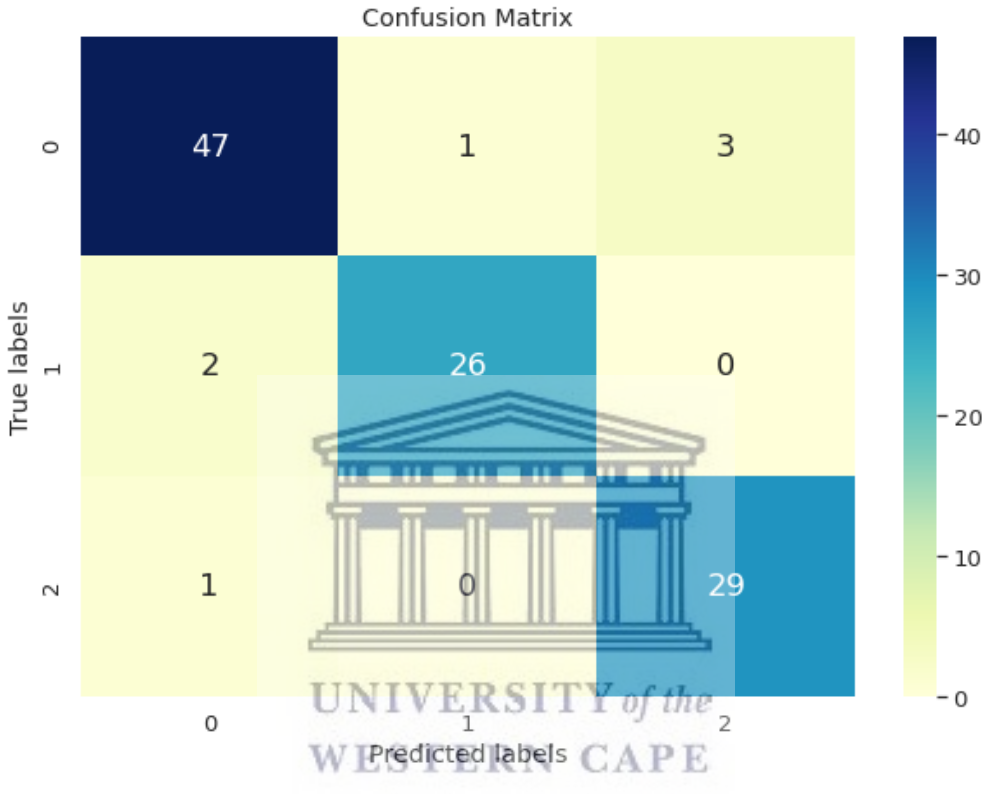


Figure 5.11: 3×3 Confusion matrix using K_Means clustering for case 3a, were baselines are clustered in to 3 groups namely 0,1 and 2. This is a really good clustering for this dataset since most of the predicted labels match the actual labels (most values are in the diagonal-pattern).

Case 4a: Ellipticity and rotation of the primary beam by 1%

This case still encounters the same problem we had for case 3a, where the stability of the confusion matrix and the accuracy changes every time we run the clustering algorithm for $k \geq 4$. While for k=3 we get the highest accuracy compared to the other k values, where the change in the confusion matrix is insignificant because we get an accuracy of $92\,\% \pm 1\,\%$. The classification report shown below summarises

the performance of the algorithm:

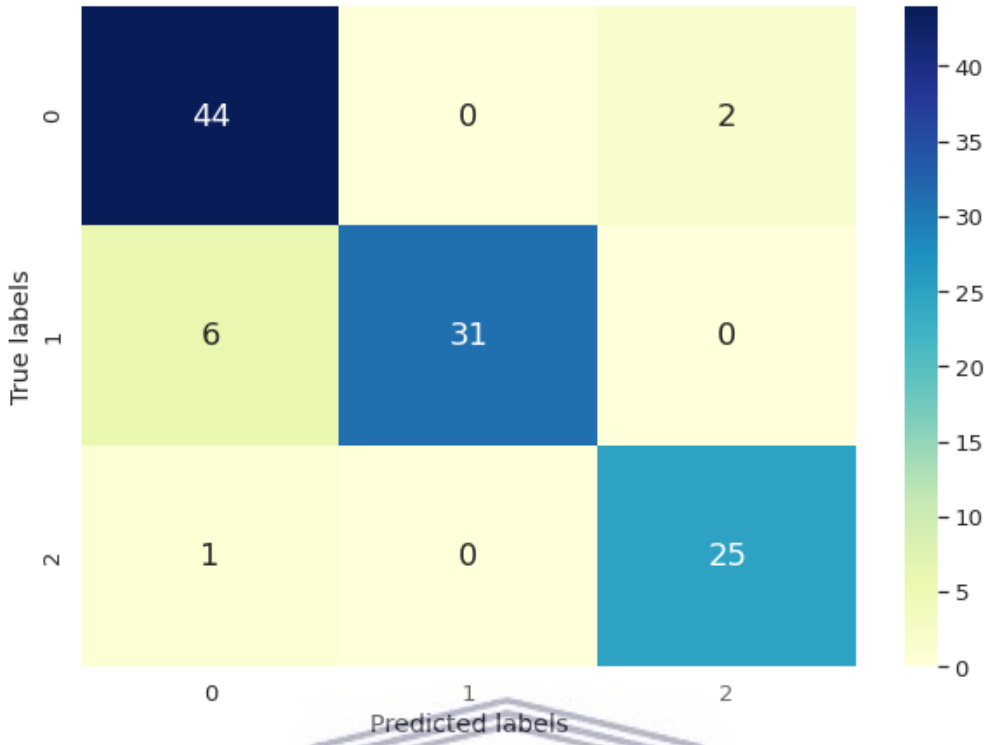


Figure 5.12: 3×3 Confusion matrix using K_Means clustering for case 4a, were baselines are clustered in to 3 groups namely 0,1 and 2. This is a really good clustering for this dataset since most of the predicted labels match the actual labels (most values are in the diagonal-pattern).

Case 4b: Ellipticity and rotation of the primary beam by 2%

This case is a little different from cases 3a and 4a in the sense that there is no stable classification for any value of k. That being said, the only k-values that seem to be more stable, are the ones for k=3 and k=4 with the accuracy of 92% and 90% respectively. The instability of the classification for both these k-values comes when 1 or 2 baselines are clustered into groups that they do not belong to. This simply means the classification alternate between two confusion matrices, but since the difference is so small, the overall accuracy of these two classifications for both k-values remains the same.

Now moving to higher k-values we encounter the same problem we had for case 3a and 4a, where the stability of the confusion matrix and the accuracy keeps on

changing every time we run the clustering algorithm for $k \geq 5$. For this reason, we did not record the accuracy for these k-values since the change in the accuracy is so unstable that it can differ by a factor of 40% for the same k-value.

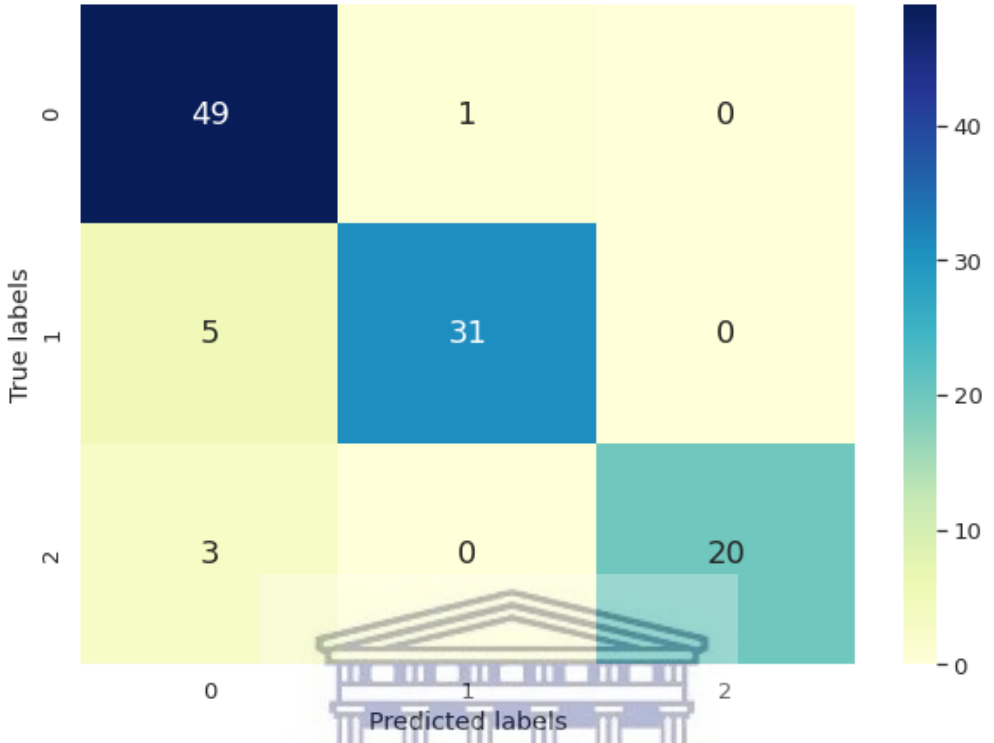


Figure 5.13: 3×3 Confusion matrix using K_Means clustering for case 4b, were baselines are clustered in to 3 groups namely 0,1 and 2. This is a really good clustering for this dataset since most of the predicted labels match the actual labels (most values are in the diagonal-pattern).

Case 5: Stretched baseline length by 10 cm

The confusion matrix for this case is a little hard to pin down because there is no stable classification for any value of k. It may be due to the nature of the non-redundancy introduced into the system since all the previous cases presented before were primary beam-dependent non-redundancies, while case 5 is a positional non-redundancy. However, k=3 is the only k-value that seems more stable with an accuracy ranging between 92% and 96%. This is a 4% difference in accuracy. One thing to make note of is that the true labels (i.e. data from case2) do not change every time we run the algorithm. The only changes are from the predicted labels which are understandable, given how some of the visibilities from other baselines

are intermingling with each other, because of added noise and gains.

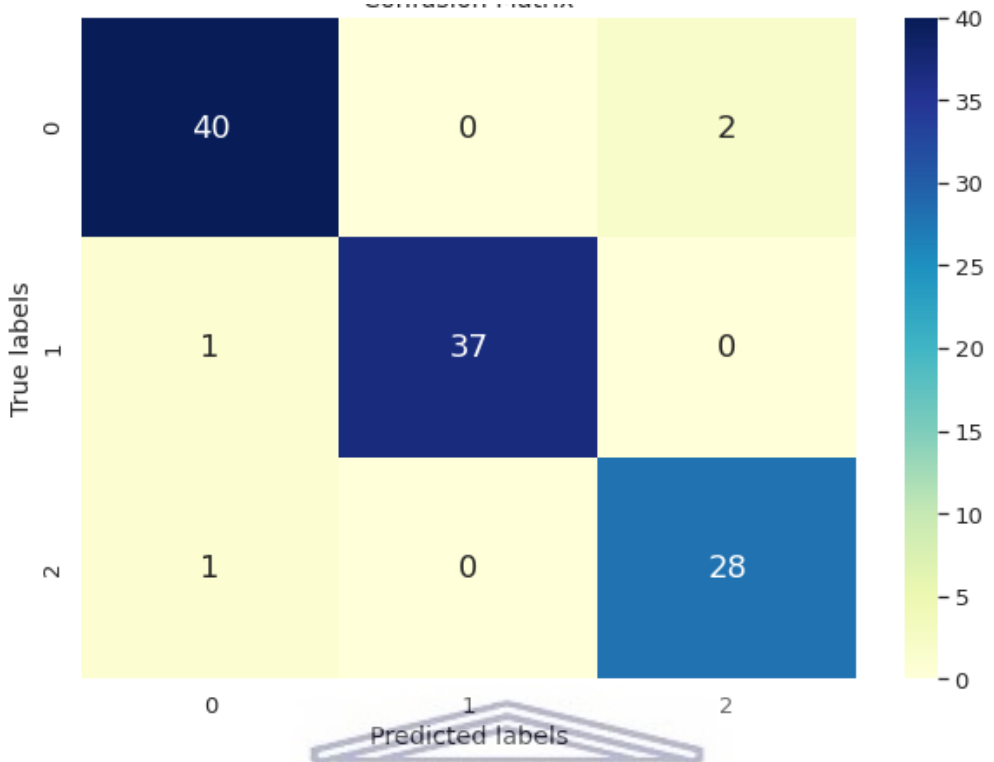


Figure 5.14: 3×3 Confusion matrix using K_Means clustering for case 5, were baselines are clustered in to 3 groups namely 0,1 and 2. This is a really good clustering for this dataset since most of the predicted labels match the actual labels. In this plot we show the matrix that gives us the highest accuracy for the same k-value

K-Means Summary

The figures in 5.15 show a classification report, which acts as a measure of the quality of predictions from a classification algorithm. But since K-means is a clustering algorithm, for this validation stage of the project, we treated K-means as a classification algorithm. As mentioned before, we used K-means to cluster the true data, therefore the results become our labels, turning this into a classification problem (this could only be done with a simulation since we know the expected results). From the results, the precision and accuracy of the model take more precedence than the other scores, because those clustered baselines are used for calibration. It requires that the baselines with similar visibility solutions be clustered together (because they are modelled to have the same visibility).

Figure 5.15 shows some further information regarding the performance of the

k-means clustering algorithm for the different cases it was tested on. This report has these statistics, precision, recall, f1-score and support. Precision counts the percentage that is correctly predicted, and recall gives the percentage baseline that the model managed to find that belongs to that group. The f1-score is the combination (the harmonic mean) of precision and recall metrics, giving us the measure of a model's accuracy on a dataset. Support is the number of baselines that are present in that cluster.

Case_3a

	precision	recall	f1-score	support
0	0.94	0.92	0.93	51
1	0.91	0.97	0.94	30
2	0.96	0.93	0.95	28
accuracy			0.94	109
macro avg	0.94	0.94	0.94	109
weighted avg	0.94	0.94	0.94	109

Case_4a

	precision	recall	f1-score	support
0	0.86	0.96	0.91	46
1	1.00	0.84	0.91	37
2	0.93	0.96	0.94	26
accuracy			0.92	109
macro avg	0.93	0.92	0.92	109
weighted avg	0.92	0.92	0.92	109

Case_4b

	precision	recall	f1-score	support
0	0.86	0.98	0.92	50
1	0.97	0.86	0.91	36
2	1.00	0.87	0.93	23
accuracy			0.92	109
macro avg	0.94	0.90	0.92	109
weighted avg	0.93	0.92	0.92	109

Case_5

	precision	recall	f1-score	support
0	0.95	0.95	0.95	42
1	1.00	0.97	0.99	38
2	0.93	0.97	0.95	29
accuracy			0.96	109
macro avg	0.96	0.96	0.96	109
weighted avg	0.96	0.96	0.96	109

Figure 5.15: Classification Report using K_Means clustering. Using K-means the optimal k-value is 3 for all cases of non-redundancy. When looking at the support column, the number 109 is the number of baselines in the group we clustered, with the other three numbers showing how many baselines are clustered in one group. This report also shows the recall, precision and f1-score for each clustered group, the overall accuracy of K-means clustering, in the range of $92 - 96\%$.

K-mean is the easiest clustering algorithm to understand and implement. For all the different cases of non-redundancy, the algorithm performs very well, giving us an overall accuracy ranging from $92 - 96\%$, which is very impressive. The only downside of k-means clustering is that the classifications change every time for some k-values, and it is only stable for selected k-values. The only k-value that seems to give good results in all the cases of no-redundancy is $k = 3$. Even though $k = 3$ is the most stable k-value, there are some levels in its stability, for instance in terms

of percentage differences case3a, case4a and case4b all have a percentage difference of 1%, while it is 4.26% for case5. So even though the difference for case5 does not seem that significant, baselines changing the groups they are clustered into could have a significant effect on the simultaneous equation for solving the gain solution, thereby affecting the resulting degree of freedom of those gain solutions.

5.4.3 Agglomerative Hierarchical Clustering (AHC)

Case 3a: Stretching the primary beam by 1%

For AHC we do not encounter the same problem as KMeans clustering, where our classifications change every time we run the algorithms for every number of clusters (for k=3 to k=7). The only cluster number giving us higher accuracy is for $k = 5$, with an accuracy of 80%. Since the AHC does not run into the same problem as KMeans, we can record the accuracy for each cluster number. For k=3, 4, 5, 6, 7 we have accuracy = 24%, 24%, 80%, 67%, 64% respectively. The classification report shows the performance summary of the best optimal cluster number.

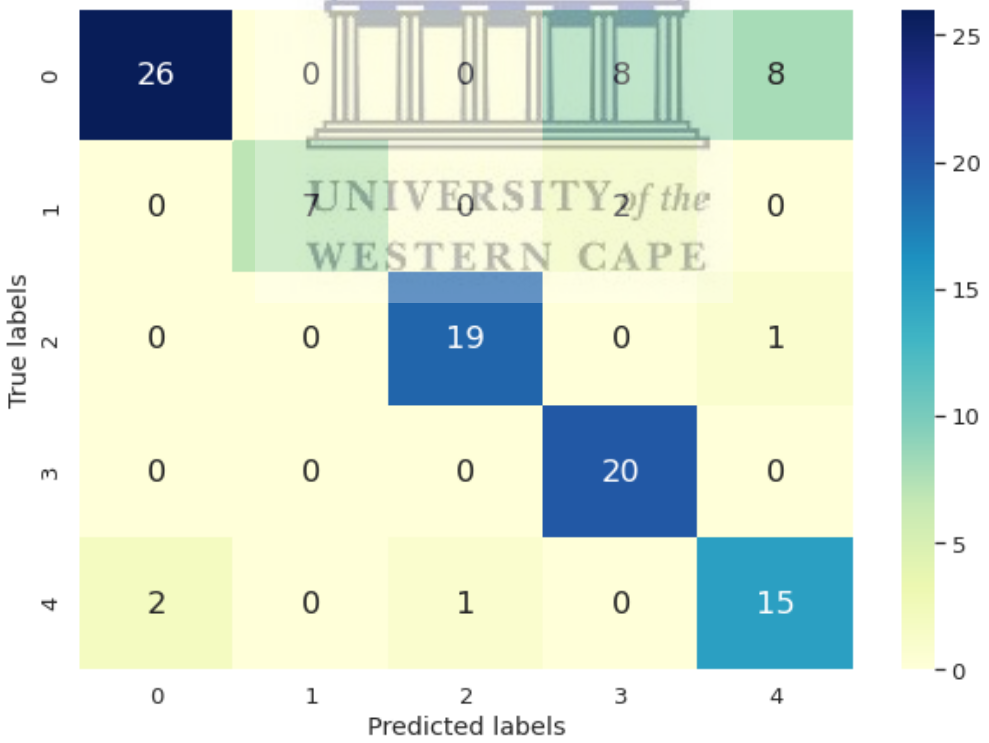


Figure 5.16: 5×5 Confusion matrix using Agglomerative Hierarchical Clustering clustering for case 3a, where baselines are clustered in to 5 groups namely 0,1,2,3 and 4.

Case 4a: Ellipticity and rotation of the primary beam by 1%

For this case, we ran the algorithms for a reasonable number of clusters, i.e. k=3 to k=7. Unlike case 3a, the only cluster number giving us higher accuracy is for $k = 3$, with an accuracy of 84%. The recorded the accuracy for each cluster number k=3, 4, 5, 6, 7 we have accuracy = 84%, 83%, 73%, 66%, 61% respectively. Unlike case 3a, the accuracy seems to follow a pattern where the accuracy decreases as the number of clusters increases.

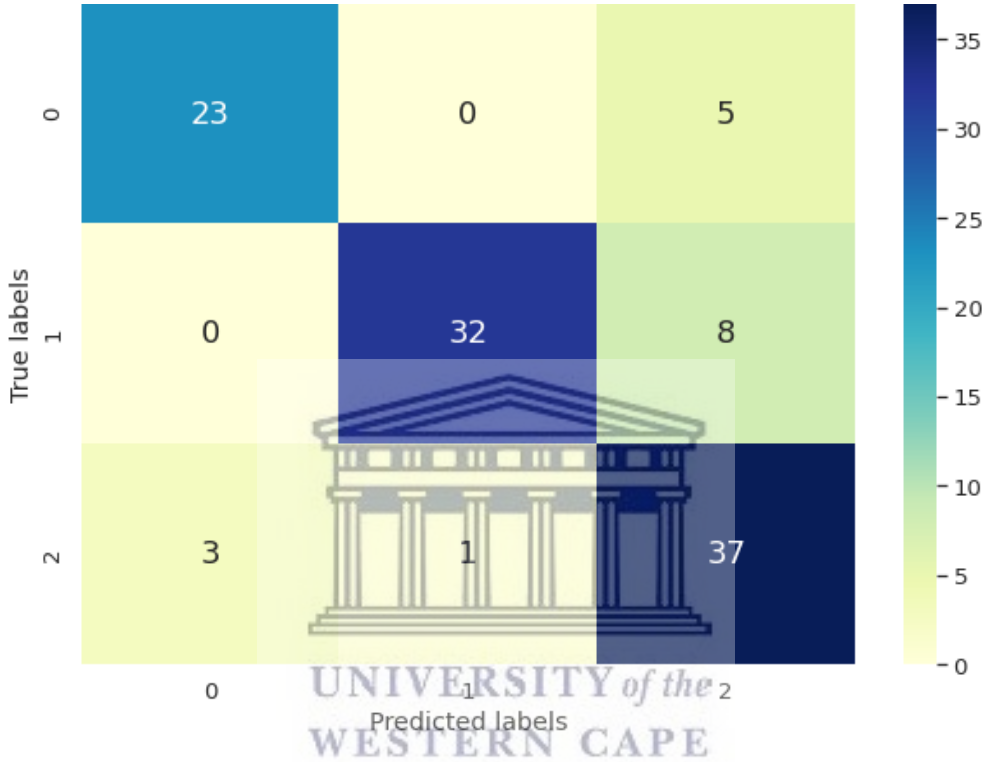


Figure 5.17: 3×3 Confusion matrix using Agglomerative Hierarchical Clustering clustering for case 4a, where baselines are clustered in to 3 groups namely 0,1 and 2.

Case 4b: Ellipticity and rotation of the primary beam by 2%

Just like in the previous case, we run the algorithms for k=3 to k=7. For case4b the only cluster number giving us higher accuracy is for $k = 4$, with an accuracy of 82%. The accuracy for each cluster number k=3, 4, 5, 6, 7 we have the accuracy = 61%, 82%, 34%, 48%, 40% respectively.

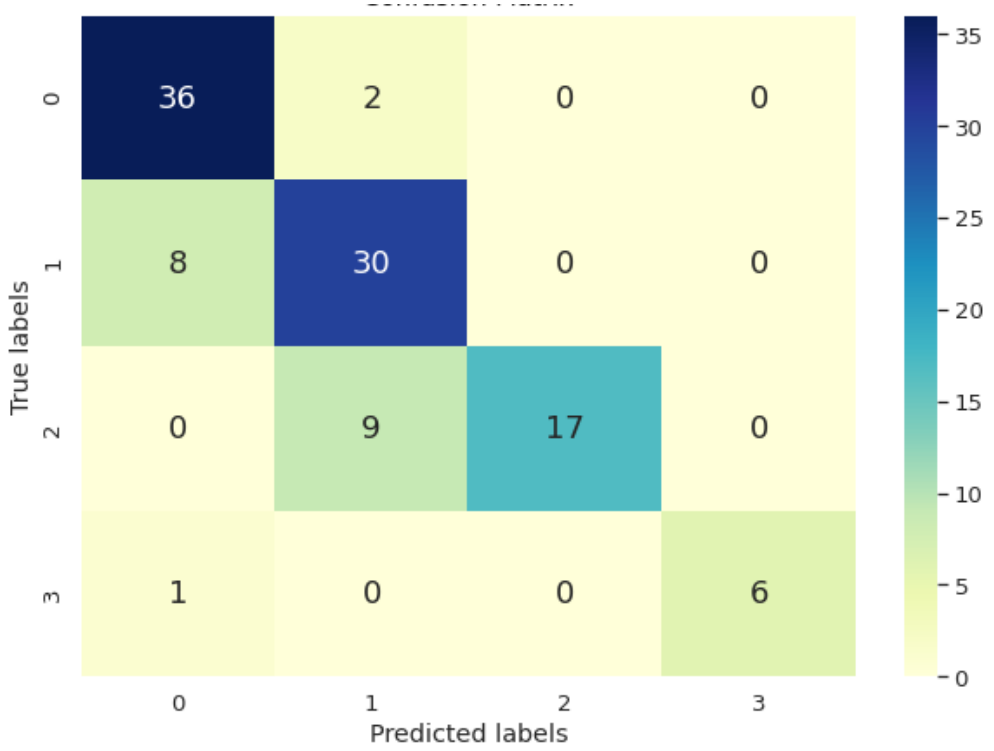


Figure 5.18: 4×4 Confusion matrix using Agglomerative Hierarchical Clustering clustering for case 4b, where baselines are clustered in to 4 groups namely 0,1,2 and 3.

Case 5: Stretched baseline length by $10cm$

For AHC we have the most stable classification since it does not change every time we run the algorithms for every number of clusters (for k=3 to k=7). For this case of non-redundancy, $k = 3$ gives us the highest accuracy of 85% for any k-value. The accuracy for each cluster number k=3, 4, 5, 6, 7 we have accuracy = 85%, 53%, 81%, 18%, 70% respectively.

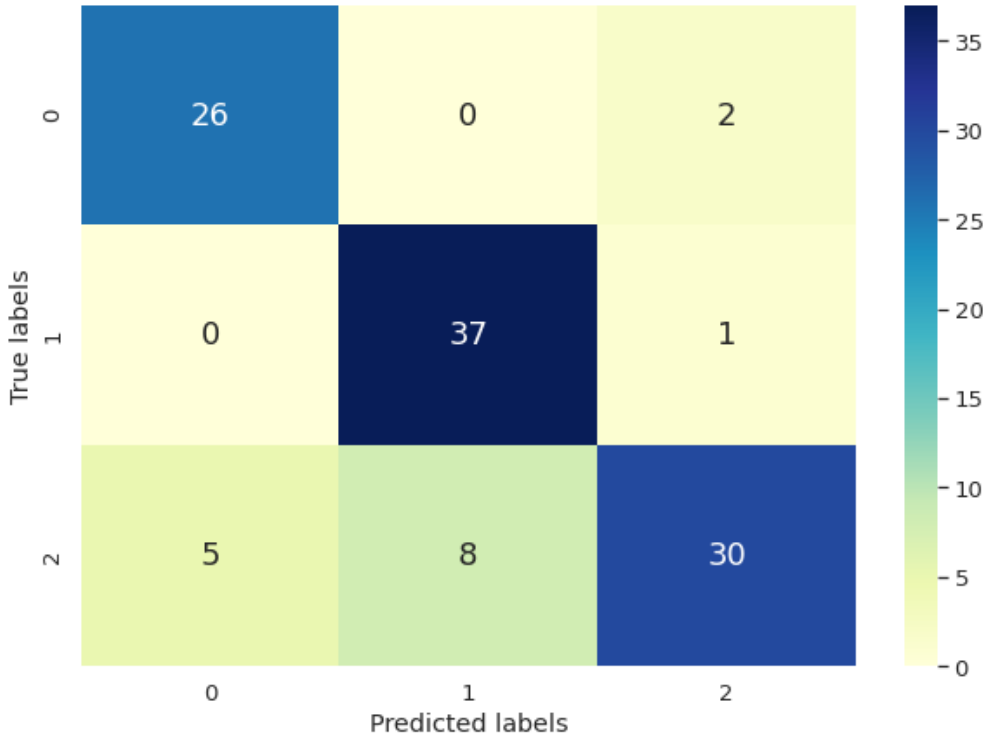


Figure 5.19: 3×3 Confusion matrix using Agglomerative Hierarchical Clustering clustering for case 5, where baselines are clustered in to 3 groups namely 0,1 and 2.

AHC Summary

The Agglomerative Hierarchical Clustering algorithm still produces good results given the dataset we are using and the number of features supplied. As explained in section 3.3, AHC does not suffer the same flaw as k-means clustering of changing output every iteration (thereby changing the accuracy). This advantage offers us a new perspective on how we interpret the performance of this algorithm and also enlightens us on the sensitivity of choosing the right k-value. It allows us to display the accuracy per case for each k-value. From these accuracy values, we can see that all the other cases of non-redundancy show no trend on how k affects the accuracy of our model, except for case4a. For case4a, the accuracy decreases as the value of k increases. Meaning that for this case, the more we split the baselines into even more subgroups, the less accurate the model is in separating baselines based on their visibility solutions. As for the other cases, we see no such trend in making AHC less reliable for our case study since we could not have an intuition on what the k-value is since it changes for each case. By saying AHC is less reliable, we mean in terms

of low accuracy compared to k-means, but more reliable in terms of reproducibility.

Case_3a

	precision	recall	f1-score	support
0	0.93	0.62	0.74	42
1	1.00	0.78	0.88	9
2	0.95	0.95	0.95	20
3	0.67	1.00	0.80	20
4	0.62	0.83	0.71	18
accuracy			0.80	109
macro avg	0.83	0.84	0.82	109
weighted avg	0.84	0.80	0.80	109

Case_4a

	precision	recall	f1-score	support
0	0.88	0.82	0.85	28
1	0.97	0.80	0.88	40
2	0.74	0.90	0.81	41
accuracy			0.84	109
macro avg	0.86	0.84	0.85	109
weighted avg	0.86	0.84	0.85	109

Case_4b

	precision	recall	f1-score	support
0	0.80	0.95	0.87	38
1	0.73	0.79	0.76	38
2	1.00	0.65	0.79	26
3	1.00	0.86	0.92	7
accuracy			0.82	109
macro avg	0.88	0.81	0.84	109
weighted avg	0.84	0.82	0.82	109

Case_5

	precision	recall	f1-score	support
0	0.84	0.93	0.88	28
1	0.82	0.97	0.89	38
2	0.91	0.70	0.79	43
accuracy			0.85	109
macro avg	0.86	0.87	0.85	109
weighted avg	0.86	0.85	0.85	109

Figure 5.20: Classification Report using Agglomerative Hierarchical Clustering. AHC does not have a universal optimal k-value since this value differs on case-to-case bases for all the cases of non-redundancy tested. The support column has the value 109, which is the number of baselines in the group we clustered, with the other three numbers showing how many baselines are clustered in one group. This report also shows the recall, precision and f1-score for each clustered baseline group. The overall accuracy of AHC is in the range of $80-85\%$.

5.4.4 Best performing Clustering algorithm

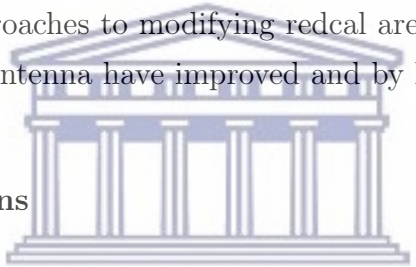
The previous two subsections show how the clustering algorithms respond to data from different cases of non-redundancies. It has shown that the clustering algorithm can produce good clusters based on the accuracy we are getting. The most notable difference between these clustering algorithms is that the final baseline clusters we get from k-means is not reproducible (unless given enough tries for our data) compared to AHC. This issue was discussed in section 3.2, which has to do with the iterative nature of k-means, which leads to different results because of discrete starting points. Even with these shortcomings, k-means still performs much better than the AHC (giving us the accuracy of $92-96\%$ compared to $80-85\%$). The iterative process of k-means does not seem to become much of an issue if there are more

baselines to cluster, given the differences in percentage changes (±1%) we get after multiple iterations. This small margin of error is equivalent to only about 1 or 2 baselines clustered into different groups.

Overall given the data, k-means is the best-performing clustering algorithm between the two. Therefore k-means is used in modifying redcal, so the results reported and discussed for the entirety of this work are done using k-means clustering.

5.5 Comparing RedCal and Logi_Cal

This section is split into different subsections that will all be centred around comparing the solutions we get from redcal and logi_cal. The first section, 5.5.1 will deal with the visibility solutions and the second section, 5.5.2 will deal with the antenna gain solutions. In all these subsections, we are trying to see if we could recover the visibility and gain solutions better when using logi_cal instead of redcal. Furthermore, different approaches to modifying redcal are discussed, based on how the gain solutions of each antenna have improved and by how much.



5.5.1 Visibility solutions

This subsection reports on two main simulation parameters; first, the simulation is a 10-antenna array run with nfreq=10, ntimes=10 and second, the simulation is a 124 antenna array run with nfreq=120, ntimes=1. These simulations have added noise and antenna gains and run with the same case of non-redundancy(case3a- Stretched primary beam).

The main focus of this subsection will be on trying to quantify the results we get from redcal and logi_cal. To do that, we compare the visibility solutions we get from these calibration methods with what is known (true_data). Since we are dealing with simulations of the big array, we cannot visually observe all the baselines to check for improvements in the calibration method, for that we employ some statistical methods to assist in our analysis. There are about 6 statistical methods we are considering namely: The directed Hausdorff distance (Birsan & Tiba, 2005; Knauer et al., 2011); Partial Curve Mapping (PCM) (Witowski & Stander, 2012); Discrete Fréchet distance (Eiter & Mannila, 1994); Dynamic Time Warping (DTW)

(Müller, 2007); Area between two curves (Raju, 1988) and Curve-Length distance metric. Ultimately all these methods share the same goal, that is to measure the similarities of these curves (shown in figures 5.21 and 5.23). However, this subsection mainly concentrates on two; Discrete Fréchet distance and Area between two curves. Because these two give very noticeable results, based on Figure 5.22, and they are very different in their approach to determining the similarities between these curves.

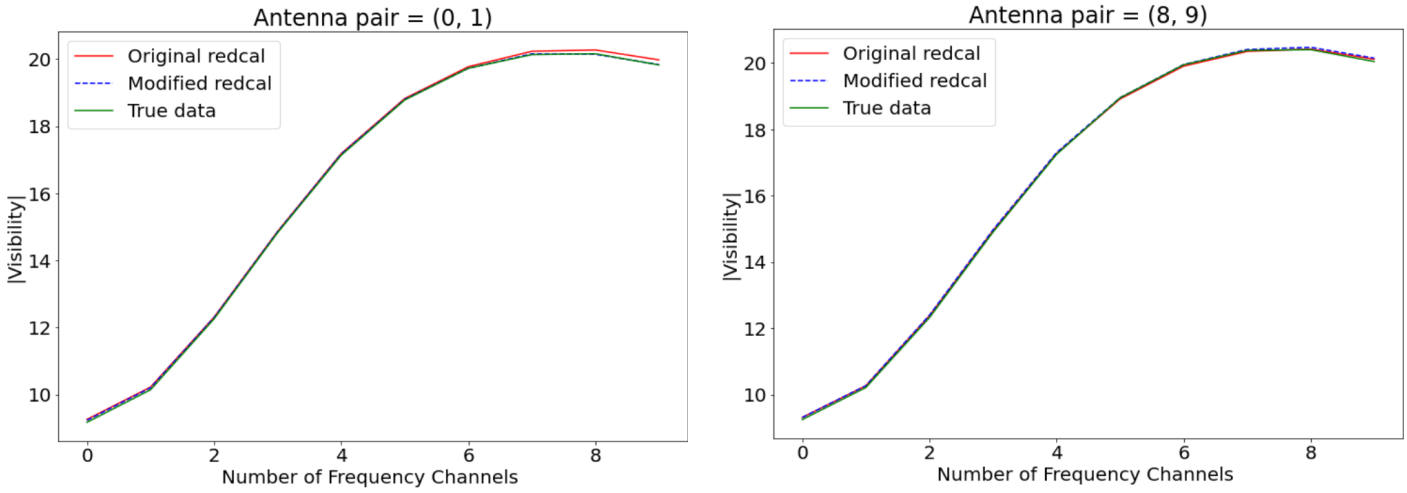
Discrete Fréchet distance: is a measure of similarity between curves, introduced by Fréchet (1906). This method can help in distinguishing the difference/similarities between two curves by taking into account the location and ordering of the points along those curves. It does this by calculating the farthest that the curves separate. It means that all other points on the curve are closer together than this distance. This method is even more ideal because it takes the continuity of the curves into account, unlike the directed Hausdorff distance, which measures the mismatch between two point sets, making it unsuited for curve matching (Witowski & Stander, 2012).

Area between two curves: As the name suggests, this method calculates the area between two curves you want to compare. The curves are similar if the area is small, but not so similar if the area is not small. Notice that we did not define "small", and for our case, it is not necessary since we are comparing the area between two pairs of curves, the first being redcal vs true_data and the second being logi_cal vs true_data. It simply means the curve pair that gives a lower area has more similarity than the other.

Simulation: N_antenna =10, nfreq=10, ntimes=10

To run calibration using logi_cal for this case we clustered the first 4 RGBs and each is clustered into 3 subgroups. In this simulation we can visually see that there is some improvement with the calibration method for one baseline and no improvement for the other baseline. We have some instances where the there is some improvements in the calibration methods and there other instances where there is none. For this particular simulation we have more improvements than not, 6 out of 7 baselines to be exact.

Since all the statistical methods are different in their approach, it means they all have different scales. To visualize the results, we plot results for Discrete Fréchet



(a) Antenna pair (0,1)

(b) Antenna pair (8,9)

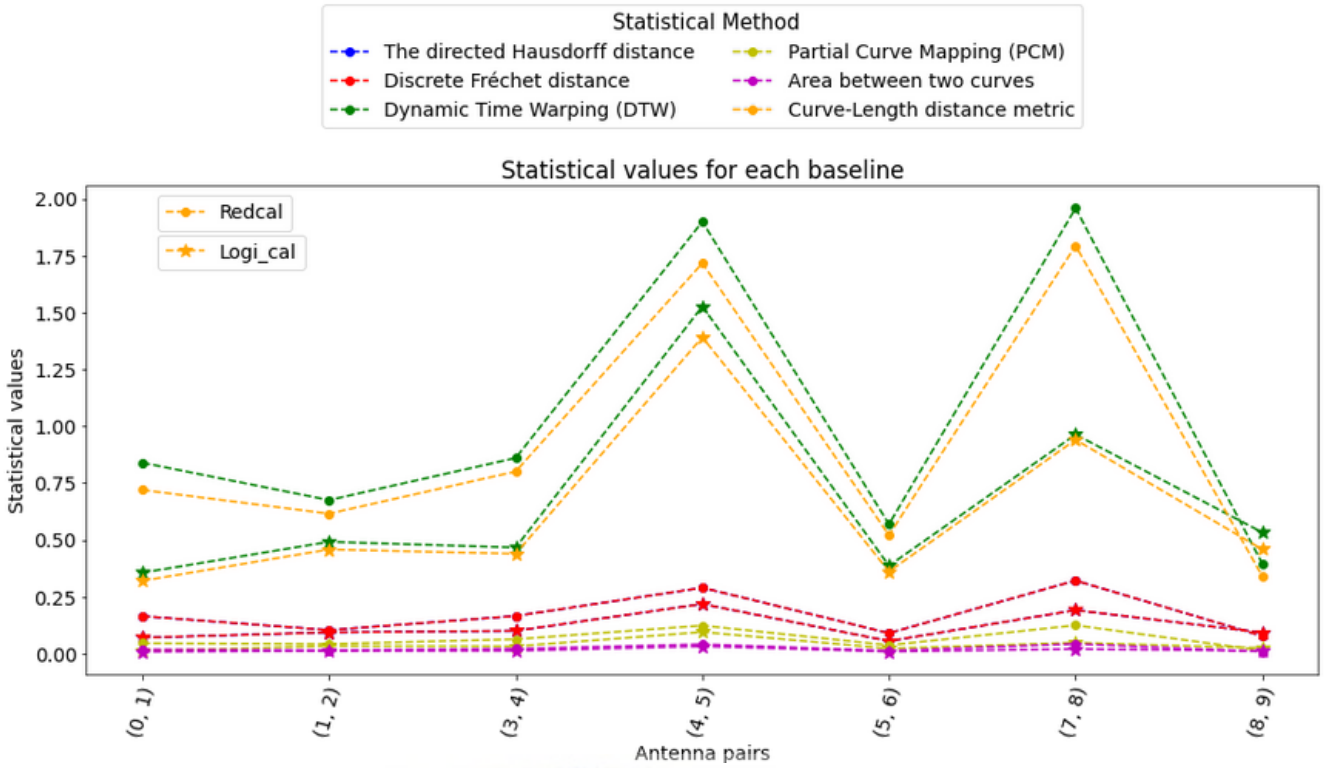
Figure 5.21: Plot for two baselines/antenna pairs in a redundant baseline group of 7 baselines. The Plot compares the visibility solutions from redcal, logi_cal and true_data (no added noise and gains).

distance and Area between two curves separately. From these plots (Figure 5.22), there are many improvements in the visibility solutions for the first 6 baselines except the last baseline, and this is shown in Figure 5.21(b).

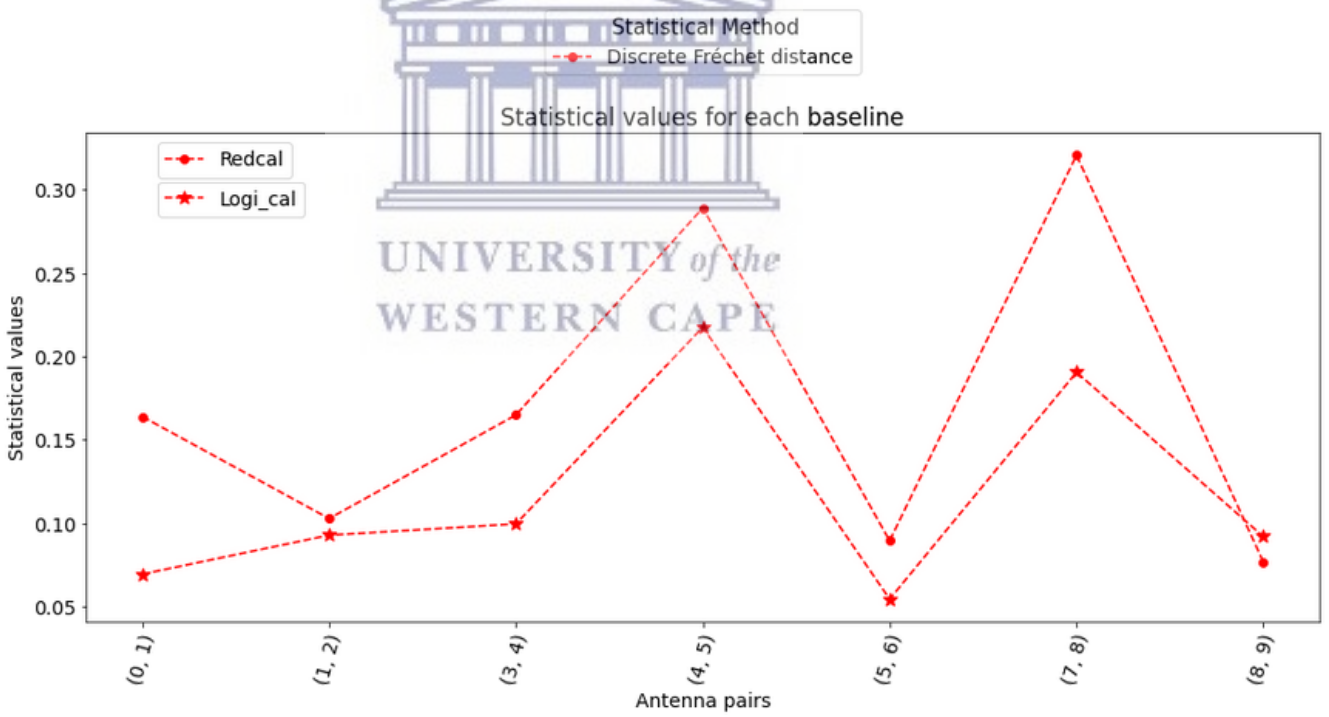
We use the gains we get from steps 7 & 8 of Figure 4.2 to give us the new visibility solutions for each calibration method and use those in our analysis. For this simulation, we cluster the first 4 redundant baseline groups. The plots in Figure 5.21 show the visibility solutions for the antenna pair (0,1) and (8,9), each of those baselines has the visibility solutions for true data, redcal and logi_cal plotted against each other. For antenna pair (0,1), we can see that the logi_cal line is superimposed over the true data line, and this means that our calibration can recover the calibration solutions better than redcal. But for antenna pairs (8,9), the performance of logi_cal cannot be determined, because all the lines are superimposed over each other. Because of this inability to gauge the difference between these calibration methods, we employ the help of some statistical methods that can compare the similarities between curves.

To do this, we used Discrete Fréchet distance and Area between two curves to calculate the similarities between the true visibilities, redcal and logi_cal visibility solutions. Even though the statistical methods differ in approach, their interpretation of the results is the same, making them easier to compare. So when comparing

two curves (the visibility solutions) for each baseline, if the values from these methods are closer to zero, the more similar those curves are. Figure 5.21(a) shows that logi_cal is improving the visibility solutions for the first six antenna pairs (out of 7), but it does not work for antenna pair (8,9), giving us a success rate of 85.7%. Figure 5.21(b) shows how logi_cal is failing to improve the visibility solutions for antenna pair (8,9), confirming the previous conclusion. With this, it is fair to say that logi_cal does improve the visibility solutions, with an 85.7% success rate for this case.



(a) All Statistical methods

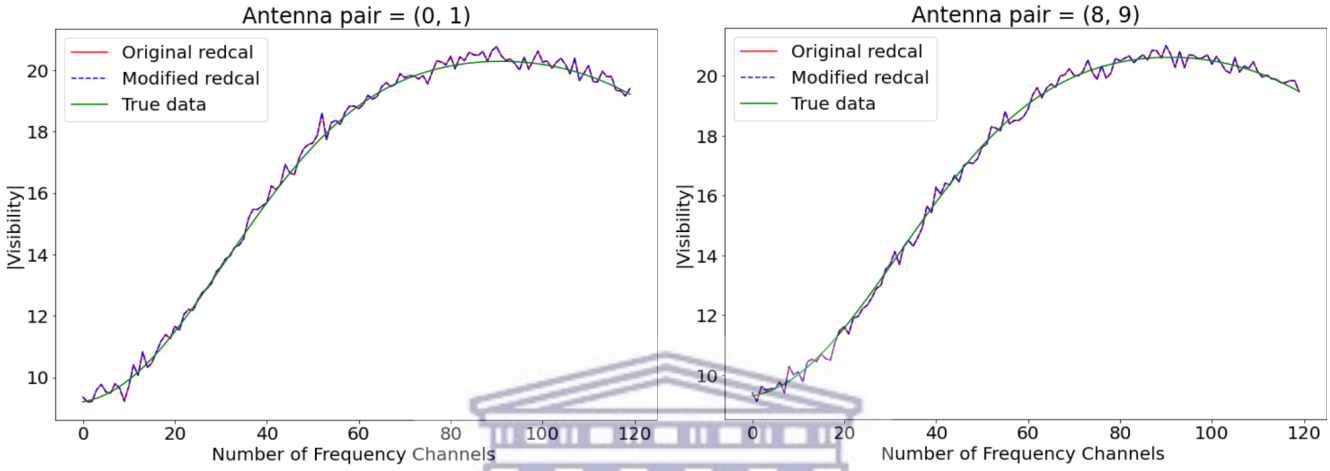


(b) Discrete Fréchet distance

Figure 5.22: Plot of statistical methods for all baselines/antenna pairs in a redundant baseline group of 7 baselines. The circle represents the statistical value for redcal and the ⋆ represents logi_cal, and the one with the lowest value per baseline is closer to the the true_data. (a) shows the results for all six statistical methods and (b) shows only for one, the Discrete Fréchet distance.

Simulation: N_antenna =124, nfreq=120, ntimes=10

To run calibration using logi_cal for this case, we clustered the first 30 RGBs and each was clustered into 3-subgroups. For this simulation, we cannot visually see any improvement with the calibration method. With the curves for redcal and logi_cal being superimposed on each other as they are, we check if we could gauge if there are any improvements made on the calibration method using the mentioned statistical methods.



(a) Antenna pair (0,1) **(b)** Antenna pair (8,9)

Figure 5.23: Plot for two baselines/antenna pairs in a redundant baseline group of 109 baselines. These are plots of the visibility solutions from redcal, logi_cal and true_data (no added noise and gains)

Unlike Figure 5.22, we plot the statistical values for the Discrete Fréchet distance and the Area between two curves in different colours to make out their differences, given how similar the curves are for redcal and logi_cal are to each other. Since there are so many baselines in the redundant baseline group for this array to clearly distinguish the values for each baseline, we produced a plot for the first 20 baselines from one of the redundant baseline groups. From this, we can see some improvements for logi_cal, however, this improvement is so small that is not that notable.

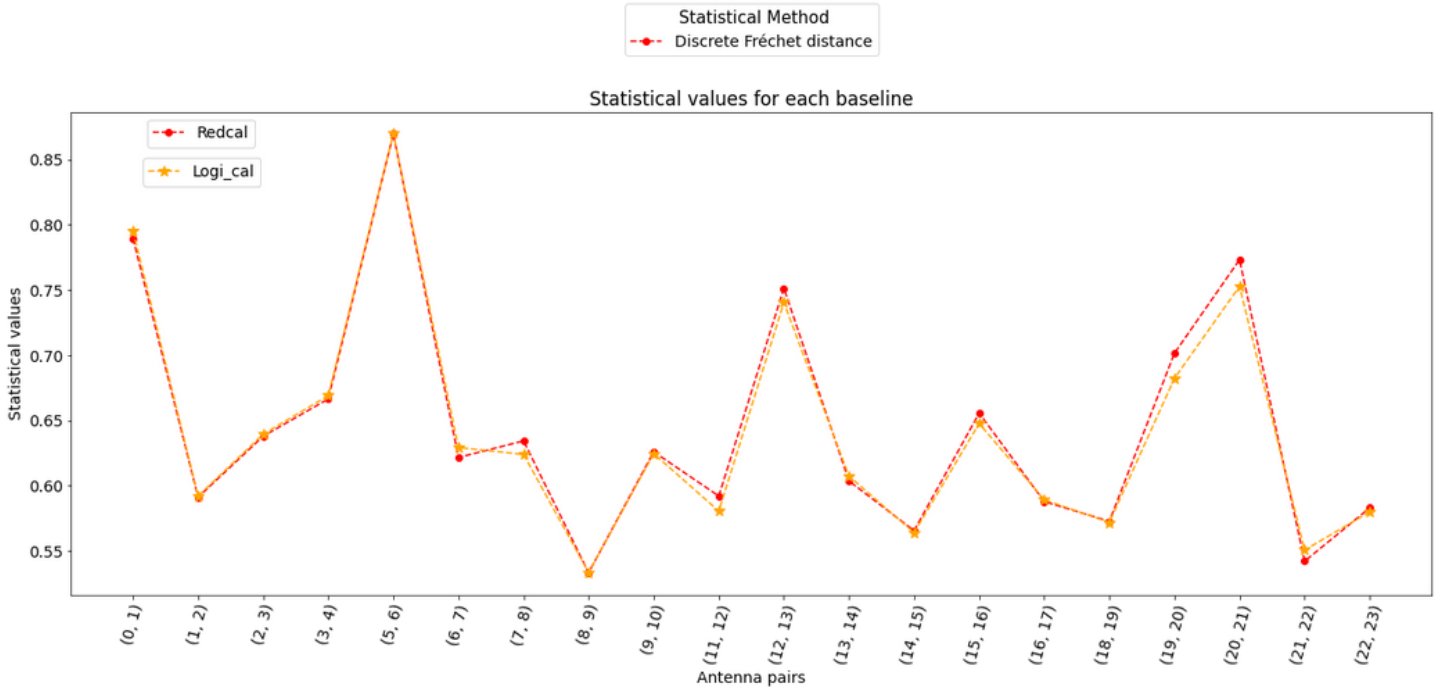


Figure 5.24: A plot of statistical methods for the first 20 baselines/antenna pairs in a redundant baseline group of 109 baselines. The circle represents the statistical value for redcal and the ⋆ represents logi_cal, and the one with the lowest value per baseline is closer to the true_data

So far, Logi_cal shows some promise since we can see some improvements in the visibility solutions we get. However, this improvement is limited to simulations with a smaller array, and we see no notable difference as we increase the array size. It is a considerable problem considering that HERA will have 350 antenna elements when complete, and up to this point, we have done only 124 antennas. Note that these are results from case3a (stretched primary beam), and since gain solutions are easier to work with, we will try to see if we can reach the same conclusion for case3a using gain solutions. If the trend is the same when comparing the gain and visibility solutions for case3a, then we would know that we can investigate the improvement logi_cal using either the gain solution or visibility solution. This would mean that there is no need to test logi_cal for other cases of non-redundancy using visibility solutions, we can just use the gain solutions for that.

5.5.2 Gain solutions

The main focus of this subsection will be on trying to quantify the results we get from redcal and logi_cal. To do that, we compare the gain solutions we get from these calibration methods with what is known (true_data). Unlike the statistics used in the previous subsection, here we used a χ^2-like statistic which we will call Δ^2, shown in Equation 5.4, to check if there is any improvement in the gain solutions for each antenna. This Δ^2 statistic was calculated using the equation $[(gain.real - gain\_true.real)^2 + (gain.imag - gain\_true.imag)^2]$ and summing over all times and frequencies. Therefore each antenna will have only one value.

$$\Delta^2 = \sum (O_i - E_i)^2 \quad , \tag{5.4}$$

where **"O"** is the observed value, that is the gains solutions we get from redcal(g_i^{RedCal}) and/or logi_cal($g_i^{Logi\_Cal}$) and **"E"**is the expected value (true gains). Therefore this equation can be written as

$$\Delta^2 = \sum \left(\left(g_{i,real} + g_{i,imag}\right)^{Cal} - \left(g_{i,real} + g_{i,imag}\right)^{true} \right)^2 \quad , \tag{5.5}$$

The values calculated using this equation enable us to compare the calibrated (either using redcal or logi_cal) gain solutions with the true gain solutions. However, these values only allow us to make comparisons per antenna, so since this statistic can be interpreted like χ^2, the antenna with a Δ^2 value closer to zero is closer to the true value/gains. Therefore given that we might have a situation where gain solutions per antenna are not improving or sometimes even getting worse when using logi_cal, we introduce another statistic which will help us in concluding about the overall performance of logi_cal, the percentage change.

$$Percentage\_change = \frac{\bar{\Delta}^2_{RedCal} - \bar{\Delta}^2_{Logi\_Cal}}{\bar{\Delta}^2_{RedCal}} \times 100\%, \tag{5.6}$$

where $\bar{\Delta}^2_{RedCal}$ is the average of the Δ^2 values of all the antennas using redcal and $\bar{\Delta}^2_{Logi\_Cal}$ is the average of the Δ^2 values of all the antennas using logi_cal for calibration. The percentage change will help in quantifying the change in the Δ^2 values from one number to another and express the change as an increase or decrease.

But equation 5.6 can also be expressed as,

Note: $g_i^{RedCal} = \left(g_{i,real} + g_{i,imag}\right)^{RedCal}$, $g_i^{LogiCal} = \left(g_{i,real} + g_{i,imag}\right)^{LogiCal}$ and $g_i^{True} = \left(g_{i,real} + g_{i,imag}\right)^{True}$

$$
\begin{aligned}
Percentage\_change &= \frac{\bar{\Delta}^2_{RedCal} - \bar{\Delta}^2_{Logi\_Cal}}{\bar{\Delta}^2_{RedCal}} \\
&= \frac{\frac{1}{N}\sum \bar{\Delta}^2_{RedCal} - \frac{1}{N}\sum \bar{\Delta}^2_{Logi\_Cal}}{\frac{1}{N}\sum \bar{\Delta}^2_{RedCal}} \\
&= \frac{\frac{1}{N}\sum\left[\sum\left(g_i^{RedCal} - g_i^{True}\right)^2\right] - \frac{1}{N}\sum\left[\sum\left(g_i^{LogiCal} - g_i^{True}\right)^2\right]}{\frac{1}{N}\sum\left[\sum\left(g_i^{RedCal} - g_i^{True}\right)^2\right]} \\
&= 1 - \frac{\frac{1}{N}\sum\left[\sum\left(g_i^{LogiCal} - g_i^{True}\right)^2\right]}{\frac{1}{N}\sum\left[\sum\left(g_i^{RedCal} - g_i^{True}\right)^2\right]}
\end{aligned}
$$

Now including the real and imaginary parts we get,

$$
Percentage\_change = \left[1 - \frac{\sum\sum\left(\left(g_{i,real} + g_{i,imag}\right)^{LogiCal} - \left(g_{i,real} + g_{i,imag}\right)^{True}\right)^2}{\sum\sum\left(\left(g_{i,real} + g_{i,imag}\right)^{RedCal} - \left(g_{i,real} + g_{i,imag}\right)^{True}\right)^2}\right] \times 100\% \tag{5.7}
$$

This subsection will be a report for several simulation parameters: namely antenna array = 10, 75 and 124, each run with nfreq=10, 60 and 120, ntimes=10 and 60. In calibrating some of the simulations, we have excluded baselines longer than $80m$ since longer baselines have an adverse effect on the spectral structure (Orosz et al., 2019). So tables that include the notation $max\_bl\_cut = 80$ have excluded

those longer baselines, and the tables without that notation calibrate using every baseline present in the array. This series of simulations have added noise and gains, and are run with different cases of non-redundancy, discussed and demonstrated in section 4.1. With a multitude of simulations like this, we will be investigating several questions regarding the performance of logi_cal:

1. What is the most favourable number of redundant baseline groups that should be used to get the best gain solutions from logi_cal?
2. What is the Optimal K-value (from K-means) that should be used for each case of non-redundancy to improve the results?
3. Is there any improvement on the overall gain solutions when using any number of these simulation parameters?
4. Overall, how is logi_cal's performance in comparison to redcal for each of the non-redundancy cases?

Finding the best hyper-parameters:

To answer these questions, we illustrate the results in a table for the multiple simulations we ran and the parameters implemented in the calibration methods. What we have noticed when running logi_cal is that the resulting gain solutions depend on the number of redundant baseline groups (RBGs) we choose to cluster and the number of subgroups we wish to cluster those groups into (the k-value in k-means clustering). The RBGs we chose to cluster are the first 15, 25, 30, 35, 40, 50 and 60 groups. It is also important to note that increasing the number of RBGs we chose to cluster increases the baseline lengths of the antenna pairs in those groups. And each of those RGBs is clustered using k-means clustering, with $k = 2, 3, 4, 5$ *and* 6, with the choice of k being highly dependent on the fact that k-means cannot cluster RBGs whose $number\ of\ elements(baselines) < k$.

When running redundant baseline calibration, we are given a list of lists of redundant baseline groups that are predetermined by the layout of HERA, based on the antenna positions and orientation. And when running logi_cal, we are splitting those redundant baselines into even more redundant baseline groups (as shown in

the stacked histogram in Figure 5.25). This histogram in 5.25 shows that each cluster has a reasonable number of baselines per RGB, with just a few exceptions. This kind of distribution is needed when running redundant baseline calibration to avoid the clusters within each RBG having fewer baselines in them, thereby rendering the gain solutions coming from those RBG clusters to be biased. It means that including longer baselines in our calibration introduces some intrinsic spectral structures because of visibility errors via gain errors (Orosz et al., 2019).

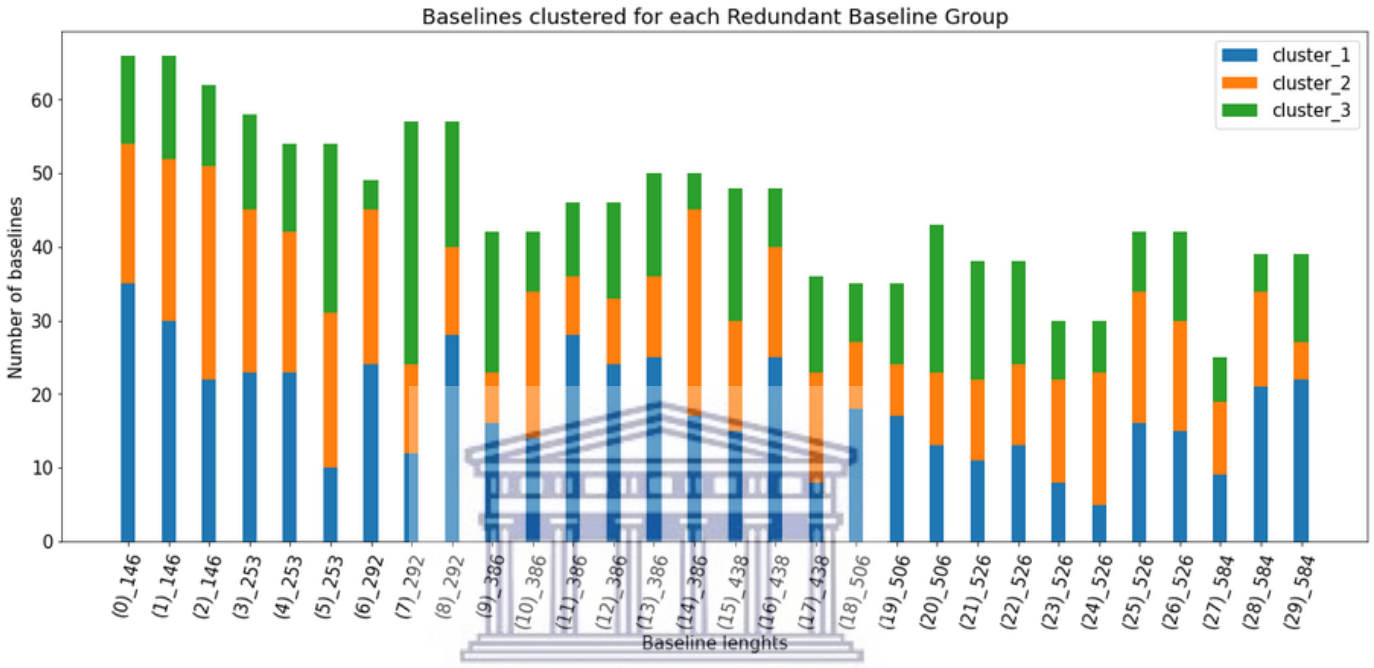
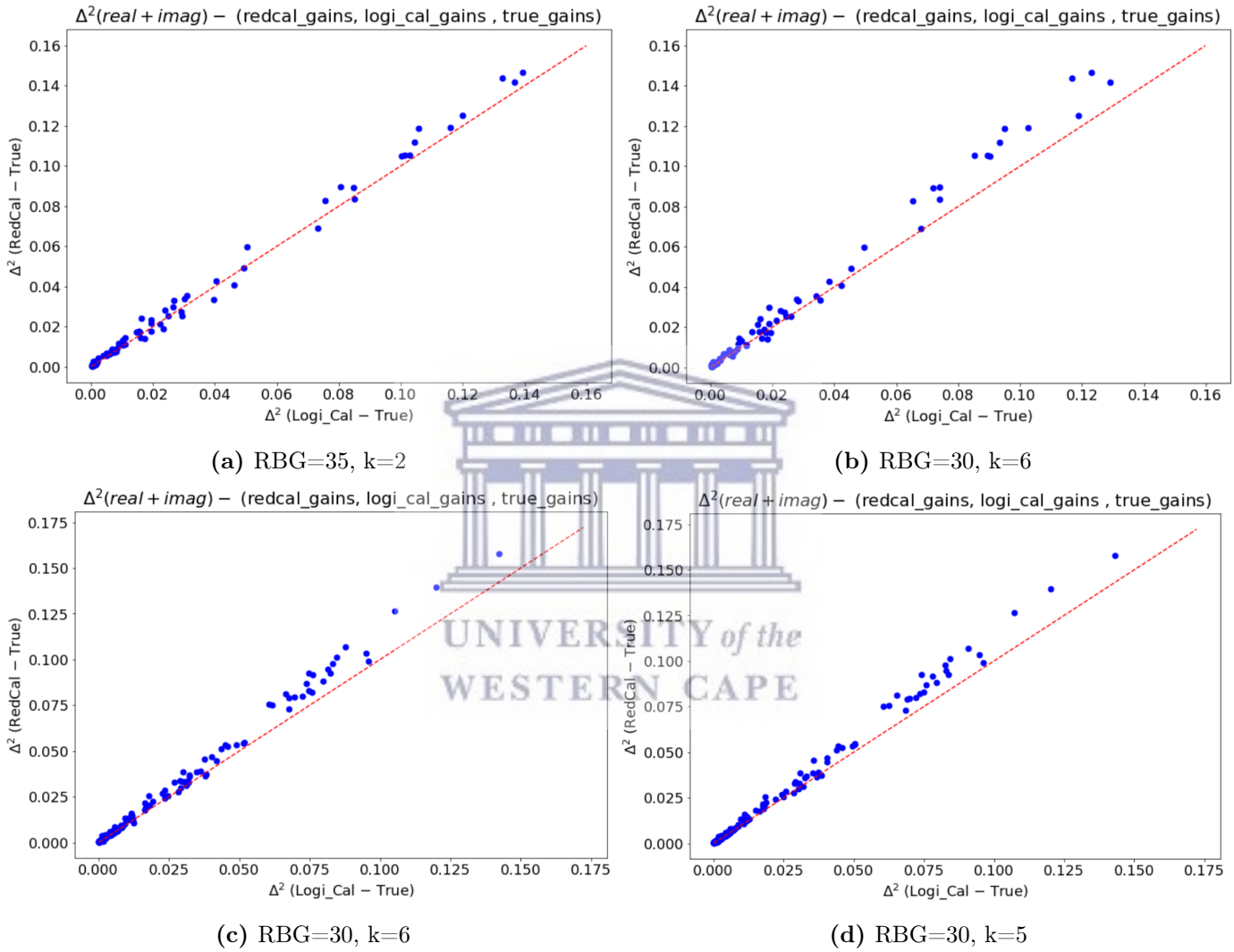


Figure 5.25: The histogram shows how the first 30 RBGs are clustered with k=3. Each of these colours represents the subgroups that the clustered baselines are grouped into. The data is from case3a with parameters: nfreq=60, ntimes=10. The x-axis is the baseline lengths corresponding to each RBG, with the numbers in the parenthesis representing the index that RBG is in. The baseline lengths are in the units $1dm$, e.g. $146 = 146\ X\ 1dm = 14.6m$.

In this subsection, we have investigated the number of redundant baseline groups that need to be split into subgroups by checking if there is an improvement in the gains solutions compared to the ones we get from redcal. These results are recorded in the table, showing the number of antennas that have improved their gain solutions based on how many RBGs have been clustered and by how many subgroups. So these antennas with improved gain solutions are taken from every antenna that is above the red line in Figure 5.26, which were calculated by finding out which antennas satisfy this condition $\Delta^2_{logi\_cal} < \Delta^2_{redcal}$. In the table, the best, second best

and third best parameters are colour coded by green, yellow and red, respectively. The colour coding is needed because the number of antennas with improved gains on these tables does not tell the whole story, mainly how large is that improvement. So instead of studying the entire image catalogue visually, by colour coding, we are leaving the best images (parameters) per simulation that we are visually checking. All the images that we are reporting in the table resemble the one shown in Figure 5.26.



(a) RBG=35, k=2 **(b)** RBG=30, k=6

(c) RBG=30, k=6 **(d)** RBG=30, k=5

Figure 5.26: Plot represents the Δ^2 values for logi_cal against redcal, where each point represents an antenna. The red diagonal line $x = y$ and every antenna above that line means that they have improved gains when using logi_cal to calibrate. The titles in each of these plots are the parameters used in logi_cal, RBG being the first number of redundant baseline groups and k is the k-value in k-means used for calibration. All these figures are for the case3a simulation run with nfreq=120, ntimes=10. Figures (a)-(b) are for an array with 75 antennas, while Figures (c)-(d) are for an array with 124 antennas

The Δ^2 values plotted in Figure 5.26 are calculated using Equation 5.4. Figures (a) and (c) are the best performing parameters based on the number of antennas above that redline. However, looking at the plots, we can see that (b) is the best performing given the number of antennas that are even more far away from the redline than in (a), but numerically (a) is the best with only a 1-antenna difference (looking at the table). It is also the reason we need a table and colour coding of the best three performing parameters per case of non-redundancy since this indicates that only going by the number of antennas that have improved gain solutions can be misrepresentative of the overall performance of logi_cal. With this, the same attention to detail given to (a)-(b) cannot be applied to (c)-(d), because unlike the former, we cannot visually see the difference between (c) and (d) meaning that in deciding the perfect parameters for calibration, the table is the best option.

N_antennas = 75, nfreq=120, ntimes=10						
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)
Case 3a	2	57	52	56	59	48
	3	50	47	50	47	35
	4	56	53	57	56	46
	5	56	49	56	55	41
	6	52	50	58	53	38
	Avg	54	50	55	54	42
Case 4a_0.01	2	45	32	36	30	24
	3	51	26	32	25	16
	4	57	32	33	28	16
	5	63	35	36	33	20
	6	63	43	43	38	23
	Avg	56	34	36	31	20
Case 4b_0.01	2	42	37	42	37	34
	3	56	35	39	31	28
	4	60	46	49	45	37
	5	58	52	53	48	33
	6	56	45	52	43	35
	Avg	54	43	47	41	33
Case 5	2	36	37	36	37	37
	3	41	38	35	38	37
	4	32	41	40	37	37
	5	40	45	46	45	46
	6	44	47	49	42	47
	Avg	39	42	41	40	41

Table 5.2: The table shows a number of antennas that have improved gains out of 75. The best three performing parameters (k-value and number of RBGs) per case of non-redundancy (case 3a, case4a, case4b and case 5) are colour-coded: green, yellow and red representing the best, second best and third best, respectively. The numbers on the far left are the k-values used in the clustering algorithm, and the numbers on top are the first RBGs clustered, including the longest-baseline length present in that group.

N_antennas = 75, nfreq=120, ntimes=10, max_bl_cut = 80						
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)
Case 1_0.05	2	-0.96	-0.67	-0.76	0.16	0.48
	3	2.15	3.39	4.12	3.53	5.62
	4	2.82	3.78	3.37	4.33	4.73
	5	2.56	3.22	3.25	-1.0	3.5
	6	4.85	1.68	0.64	-2.32	1.47
	Avg	2.28	2.28	2.12	0.94	3.16
Case 3a_0.01	2	4.58	3.75	6.57	5.53	-0.36
	3	5.32	4.13	8.44	7.1	-3.56
	4	7.42	6.53	12.16	11.49	-0.44
	5	7.92	7.15	13.37	12.28	-0.20
	6	7.63	6.93	13.56	12.83	-1.58
	Avg	6.57	5.7	10.82	9.85	-1.59
Case 4a_0.01	2	3.29	-1.82	-0.23	-1.8	-7.02
	3	3.33	-4.27	-2.52	-5.16	-13.25
	4	5.48	-2.05	0.07	-4.84	-15.09
	5	5.77	-2.48	0.8	-3.41	-14.31
	6	5.96	-1.02	2.39	-0.61	-13.02
	Avg	4.77	-2.33	0.1	-3.16	-12.54
Case 4a_0.02	2	3.42	-1.8	0.17	-1.69	-7.57
	3	3.61	-4.18	-2.71	-5.93	-13.79
	4	5.37	-2.34	-0.27	-3.98	-15.29
	5	6.17	-2.25	0.42	-4.14	-12.09
	6	6.23	0.73	2.81	-1.02	-13.66
	Avg	4.96	-1.97	0.08	-3.35	-12.46
Case 4b_0.01	2	2.93	-0.93	0.79	-1.15	-5.15
	3	4.8	-0.22	1.49	-1.48	-7.96
	4	6.26	2.57	5.58	2.33	-5.94

Continued on next page

Table 5.3 – continued from previous page

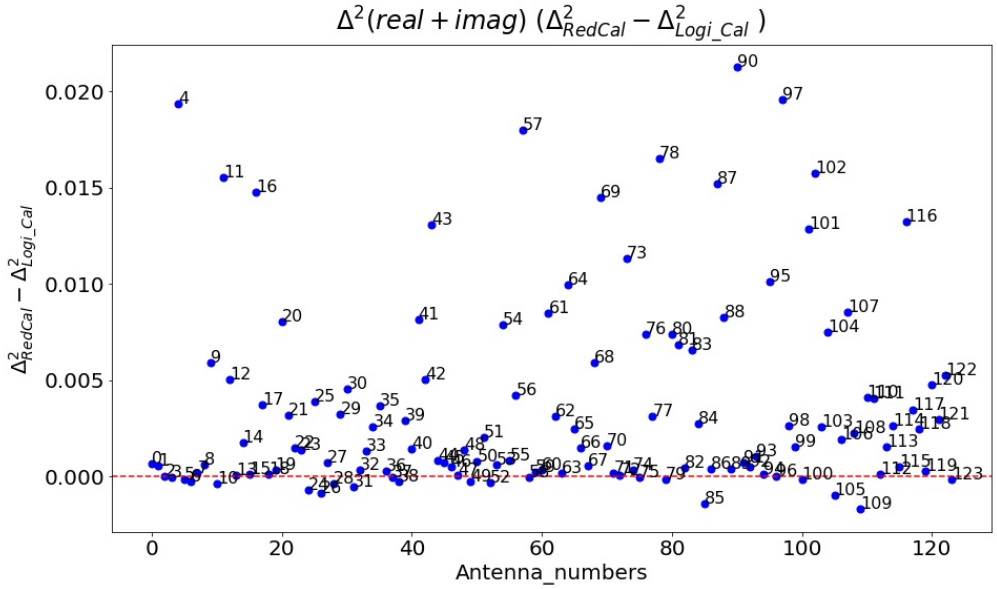
N_antennas = 75, nfreq=120, ntimes=10, max_bl_cut = 80						
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)
	5	6.83	5.07	6.86	4.65	-3.87
	6	6.69	5.28	8.73	5.08	-2.7
	Avg	5.5	2.35	4.69	1.89	-5.12
Case 4b_0.02	2	3.05	-0.26	1.4	-1.06	-4.6
	3	5.23	0.0	2.05	-1.04	-6.94
	4	6.34	3.5	5.47	1.54	-5.45
	5	6.62	4.32	8.02	4.47	-4.04
	6	6.82	5.08	6.41	4.81	-3.03
	Avg	5.61	2.58	4.67	1.74	-4.81

Table 5.3: The table shows the overall percentage change of antenna gain solutions calculated using equation 5.6. It shows the best performing parameters (k-value and number of RBGs) per case of non-redundancy (case 1, case 3a, case 4a and case 4b). The numbers on the far left are the k-values used in the clustering algorithm and the numbers on top are the first number of RBGs clustered, including the longest baseline length present in that group.

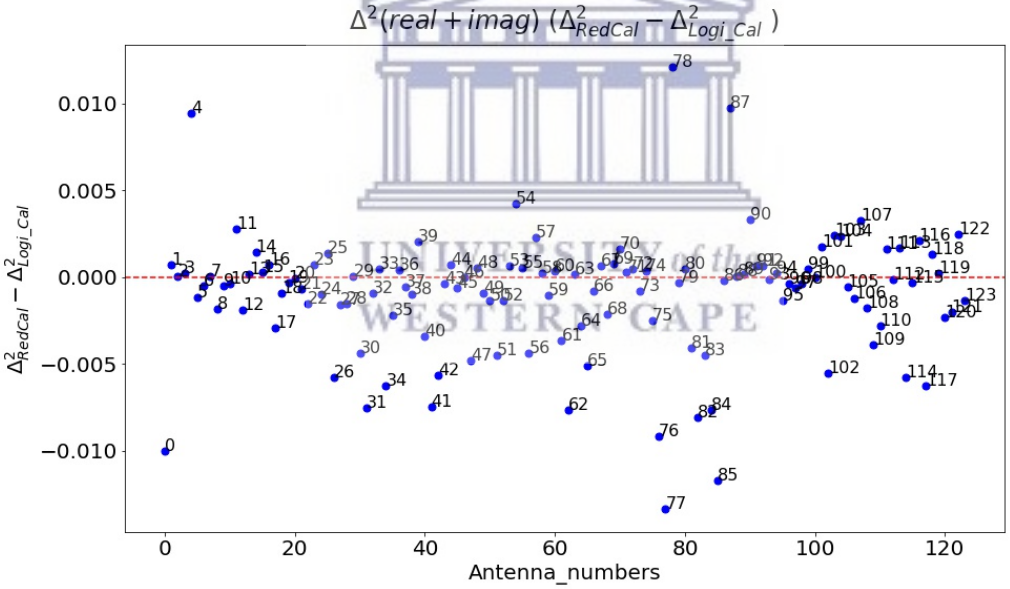
The Tables 5.2 and 5.3 act as test cases for testing the performance of logi_cal. Table 5.2 shows the number of antennas that have improved their gain solutions out of a total of 75. What is unique about this table is that when running calibration we are using all the redundant baseline groups present (so baselines of every length and orientation are included). This is not really ideal we have stated before that according to Orosz et al. (2019) longer baselines introduce some intrinsic spectral structures. That is where Table 5.3 comes in, for this table calibration is run with a $max\_bl\_cut = 80m$, this means that calibration is run using baselines that are not longer than $80m$, to reduce those spectral structure we mentioned. Because of this we can see a slight improvement in the number of antennas that have improved their gains solutions by at least $2-4$ antennas. This result is by comparing Tables 5.3 and .1 (in the appendix).

Reading more into the table it can be observed that clustering the first few baselines to run logi_cal seems to give better results, since we see a lot of overall improvement in the number of antennas with improved gains.

5.5.3 Logi_cal Option 1



(a) RBG = 30 k=6



(b) RBG = 60 k=6

Figure 5.27: Plot represents the Δ^2 values for logi_cal against redcal, with the annotated numbers representing the antenna number. (a) Shows the best performing, giving 104 antennas with improved gains and a percentage change of 12.16%. (b) Shows the worst performing, giving 52 antennas with improved gains and a percentage change of -3.5%. The red line is just the axis for $y = 0$ and every antenna above that line means that they have improved gains when using logi_cal to calibrate. RBG is the first number of redundant baseline groups and k is the k-value in k-means used for calibration

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	2	63	58	60	69	67	77	79
	3	69	70	69	72	78	90	93
	4	67	62	58	61	71	81	85
	5	62	52	56	61	67	69	91
	6	61	54	57	59	67	73	83
	Avg	64	59	60	64	70	78	86
Case 3a	2	92	84	94	88	63	76	73
	3	93	87	97	90	60	63	58
	4	94	91	104	98	62	71	65
	5	96	92	106	96	55	69	59
	6	96	90	104	98	56	67	52
	Avg	94	89	101	94	59	69	61
Case 4a_0.01	2	83	52	53	42	25	34	32
	3	81	49	51	47	27	31	32
	4	85	71	77	59	33	43	39
	5	90	67	80	64	35	45	38
	6	96	75	80	68	37	48	44
	Avg	87	63	68	56	31	40	37
Case 4a_0.02	2	84	49	54	41	25	34	31
	3	76	41	51	38	30	33	30
	4	88	63	70	58	35	44	38
	5	93	72	80	64	39	42	37
	6	97	71	86	65	40	46	42
	Avg	88	59	68	53	34	40	36
Case 4b_0.01	2	90	55	67	47	32	26	26
	3	91	65	76	62	37	31	30
	4	92	77	86	67	43	39	39
	5	101	78	90	79	40	40	34
	6	97	78	84	73	43	39	31
	Avg	94	71	81	66	39	35	32
Case 4b_0.02	2	91	59	70	49	34	29	28
	3	88	63	77	68	33	35	32
	4	96	80	83	68	40	40	36
	5	98	80	86	73	42	38	33
	6	96	77	86	74	42	40	35
	Avg	94	72	80	66	38	36	33

Table 5.4: The table shows a number of antennas that have improved gains out of 124.

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	2	0.58	-1.37	-0.73	0.39	1.04	5.41	5.63
	3	1.26	1.14	2.21	4.38	6.82	11.12	12.7
	4	-0.07	-4.59	-2.75	-0.43	0.13	9.49	11.42
	5	1.11	-6.51	-3.06	-1.0	1.02	4.41	13.59
	6	1.72	-6.05	-1.99	-0.85	-0.22	4.87	13.4
	Avg	0.92	-3.48	-1.26	0.5	1.76	7.01	11.35
Case 3a	2	5.5	4.33	6.92	5.05	0.33	2.31	1.88
	3	7.02	5.22	8.94	6.24	-1.61	-0.06	-1.54
	4	8.02	6.87	11.27	8.78	-0.84	2.31	0.76
	5	8.24	6.93	11.8	8.72	-1.67	0.73	-1.53
	6	8.24	7.07	12.16	9.24	-1.52	0.28	-3.5
	Avg	7.4	6.09	10.22	7.61	-1.06	1.11	-0.78
Case 4a_0.01	2	3.75	-1.82	-0.35	-3.54	-8.96	-10.18	-11.6
	3	4.63	-2.0	-0.01	-2.81	-10.73	-11.02	-14.86
	4	5.54	1.26	4.24	0.27	-8.31	-8.69	-11.02
	5	6.33	0.75	3.66	-0.13	-9.85	-9.94	-13.71
	6	6.91	2.04	4.73	1.63	-8.41	-7.35	-10.09
	Avg	5.43	0.05	2.45	-0.92	-9.25	-9.44	-12.26
Case 4a_0.02	2	3.25	-2.41	-0.91	-4.02	-9.56	-11.07	-12.8
	3	3.57	-3.01	-1.05	-4.64	-11.66	-13.41	-14.92
	4	5.05	0.14	2.33	-1.14	-9.7	-9.81	-13.23
	5	5.64	1.14	3.87	-0.73	-10.06	-9.86	-13.76
	6	6.79	1.16	5.24	0.84	-8.41	-9.92	-12.34
	Avg	4.86	-0.6	1.9	-1.94	-9.88	-10.69	-13.41
Case 4b_0.01	2	3.57	-1.62	0.42	-2.65	-9.16	-9.92	-11.5
	3	5.17	0.89	2.87	0.36	-8.12	-11.68	-12.91
	4	6.58	2.58	4.76	1.69	-7.24	-9.08	-12.41
	5	6.65	3.03	6.31	2.98	-7.32	-9.31	-13.54
	6	6.82	3.79	5.99	3.69	-8.51	-9.6	-15.33
	Avg	5.76	1.73	4.07	1.21	-8.07	-9.92	-13.14
Case 4b_0.02	2	3.66	-1.05	0.42	-2.34	-8.5	-9.72	-9.99
	3	5.04	0.87	2.96	0.71	-8.1	-11.01	-11.95
	4	6.24	3.08	4.48	1.40	-8.09	-9.28	-13.18
	5	6.42	3.09	5.32	2.73	-8.23	-10.76	-14.01
	6	6.52	3.84	6.69	4.05	-7.87	-9.96	-13.31
	Avg	5.58	1.97	3.97	1.31	-8.16	-10.15	-12.49

Table 5.5: The table shows the overall percentage change of antenna gain solutions calculated using equation 5.6

Both Tables 5.4 and 5.5, show the best three performing parameters (k-value and number of RBGs) per case of non-redundancy (case1_0.05, 3a, case4a_0.01, case4a_0.02, case4b_0.01 and case4_0.02). Where the best are color coded: green, yellow and red representing the best, second best and third best respectively. The second column from the left are the k-values used in the clustering algorithm and the numbers on top are the first RBGs clustered, including the longest baseline length present in that group. Now the values in the table are taken from Figure 5.27, where the number of antennas above that red line are recorded in Table 5.4 and the percentage changes for Table 5.5 are calculated using the y-axis values in that figure. Therefor each value in the cells of the tables represent the data from one figure.

The table show that there is some improvement in the overall performance of redcal, but as these tables show that improvement is highly depended on how many redundant baseline groups are clustered. For case1_0.05 clustering the first 60 redundant baseline groups gives the highest percentage change. Case3a shows that clustering the first 30 RBGs gives better results, however that improvement begins to decline as we move on clustering more RBGs. All case4 cases of non-redundancy give good results when clustering the first 15 RBGs as the performance of logi_cal begins to decline as the number of RBGs clustered are increased.

This analysis is mainly based on Table 5.5 and not table 5.4, this is because the latter just gives as the number of antennas with improved gains, but with no context on how much those gains have improved. With Table 5.5 we get almost everything regarding the performance of logi_cal when compared to redcal, because it takes into account not only the antennas with improved gains but also the antennas that have gains that have worsened. The way the percentage values are calculated is also a double-edged sword. For instance we could have a certain number of antennas with gains that have improved drastically, the values of those gains could increase the overall percentage change, thereby affecting how we report on the performance of logi_cal. However the opposite is also true.

This is why we need Table 5.4 as a reference when talking about the overall performance of logi_cal. Because by looking at the number of antennas with improved gains we could extrapolate what the percentage change should be, since we also know the number of antennas for the entire array. For example if we were to find a case where fewer antennas have improved gains but a higher percentage change we would surmise that those antennas' gains have drastically improved, therefore

giving higher percentage, which is not representative of the performance of logi_cal for the entire array. However looking at the tables we do not seem to encounter that situation.

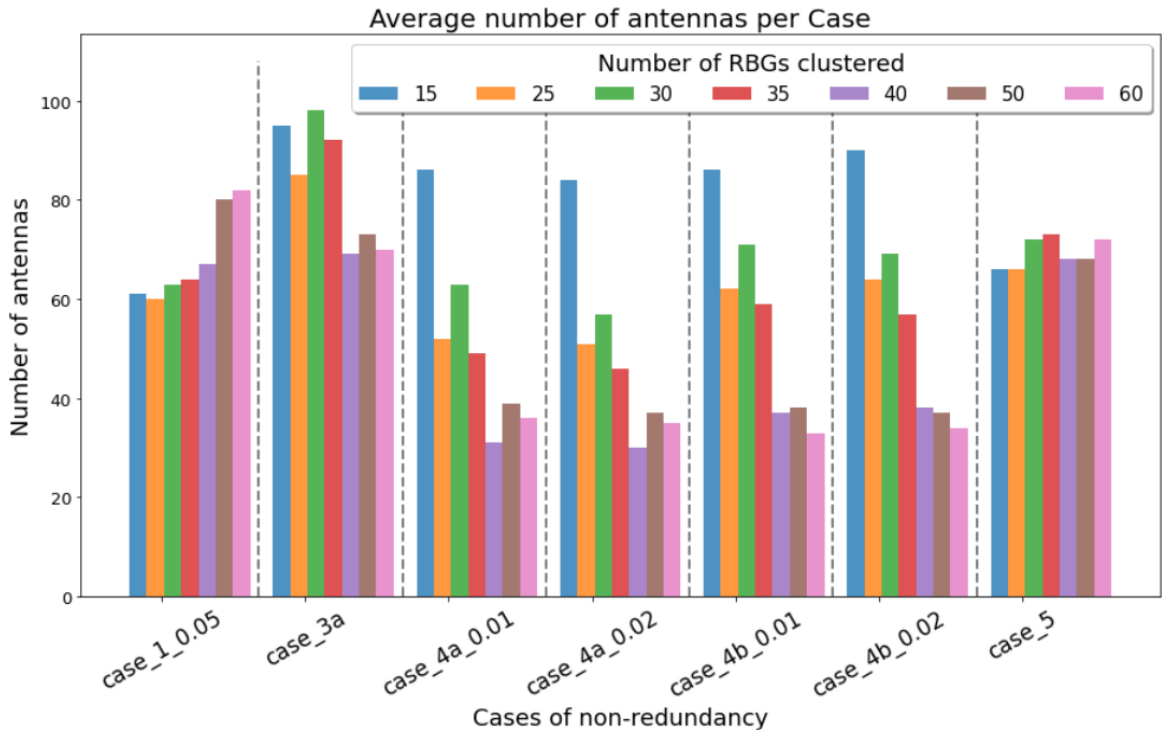
As mentioned in the conclusion for section 5.5.1, the improvement of logi_cal decreases as the size of the array increases. Now using the gain solutions, we want to see if we reach the same conclusion we did when using visibility solutions, for case3a. The best way to do this is by comparing the results from Tables 5.3 (HERA75) and 5.5 (HERA124), for case3a. From these tables we can see that the highest average improvement in case3a for HERA75 is 10.82% while for HERA124 it is 10.22%, showing that the improvement of logi_cal decreases as the array size increases.

5.5.4 Logi_cal Option 2

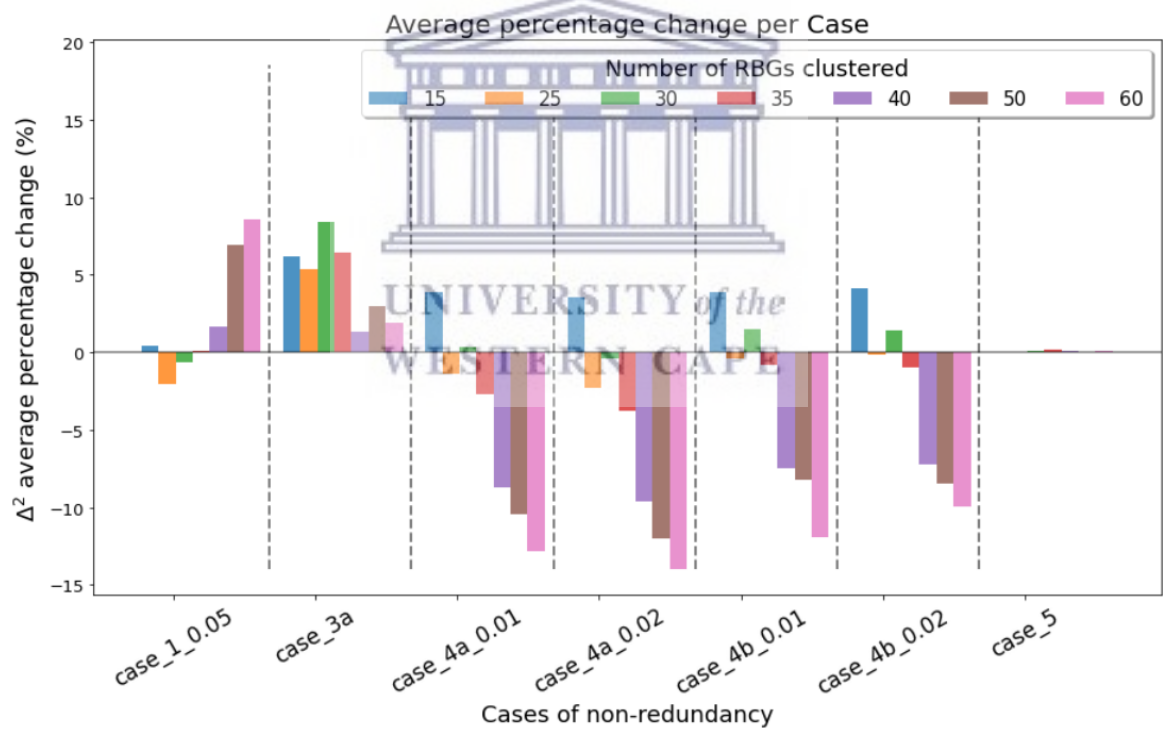
In this sub-section and the next coming sub-sections, we summarise the results of the simulations (for each Logi_cal option) like what is shown in the tables (5.4 and 5.5) using histograms. These histograms (Figure 5.28, Figure 5.29, Figure 5.30, and Figure 5.31) are a visual representation of the performance of the new calibration method (logi_cal). These plots are done to at least alleviate any confusion that might be caused by the tables referenced. The bars on these histograms represent the average values recorded on the table. For Figure 5.28(a), the y-axis is the average number of antennas that have improved their gain solutions and for Figure 5.28(b), the y-axis is the average percentage change, where positive numbers show how much our calibration method has improved the gain solutions, while the negative numbers show the opposite. On the x-axis, we have the cases of now redundancy we considered for this work and on the legends, we have the number of redundant baseline groups clustered. The blue bar (15) means we have clustered the first 15 RBGs, for the orange bar (25) means we have clustered the first 25 RBGs and so on.

The figures discussed in this subsection are for the method described in section 4.4.2. For Figure 5.28 the k-values range from [2,6], this means that in equal portions, for a given number of RBGs those are split into different subgroups. For example for 60 specified RBGs the groups could be split into five portions, 0-12 will use k=6, 12-24 will use k=5, 24-36 will use k=4, 36-48 will use k=3, and 48-60 will use k=2. While for Figure 5.29 k-values range from [4,6], then for 60 specified RBGs the groups could be split into three portions, 0-20 will use k=6, 20-40 will use k=5

and 40-60 will use k=4. For both these figures it shows that whatever the range is, the difference in improvement is very minimal, indicating that splitting shorter baselines into more subgroups and longer baselines into fewer subgroups has little impact on the overall performance of logi_cal. These extra steps will only add more unnecessary steps to the calibration method, which will only increase the time it takes to run the logi_cal.

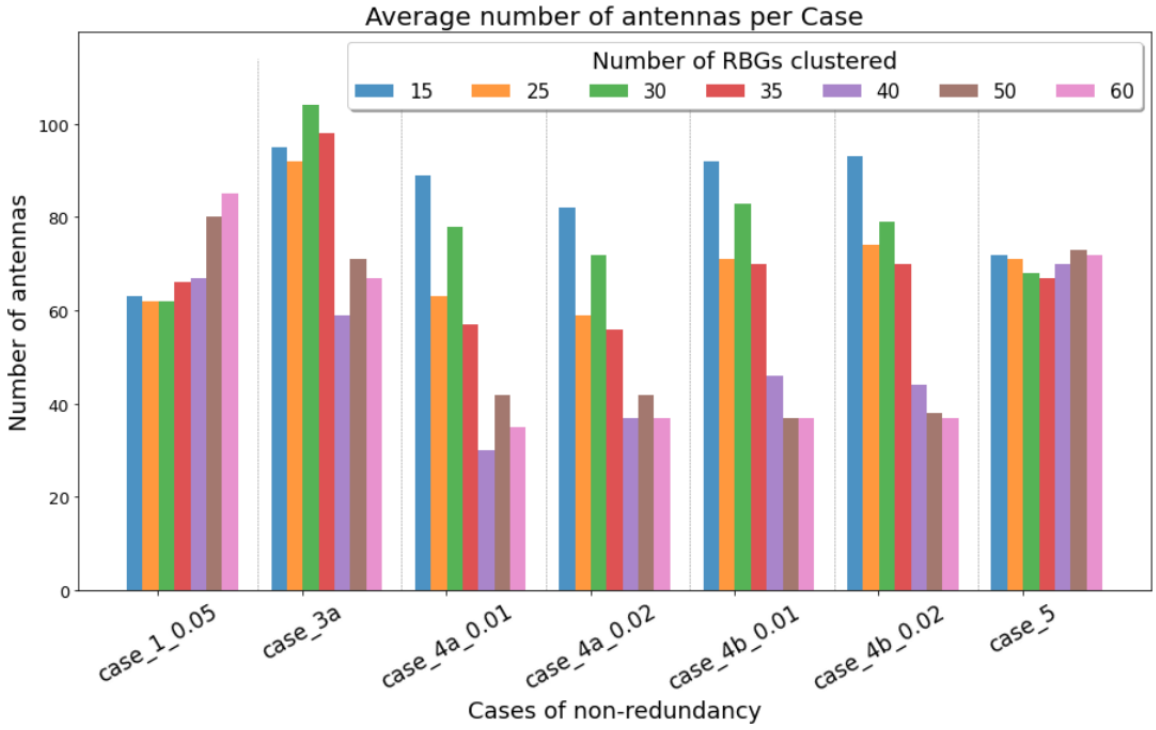


(a)

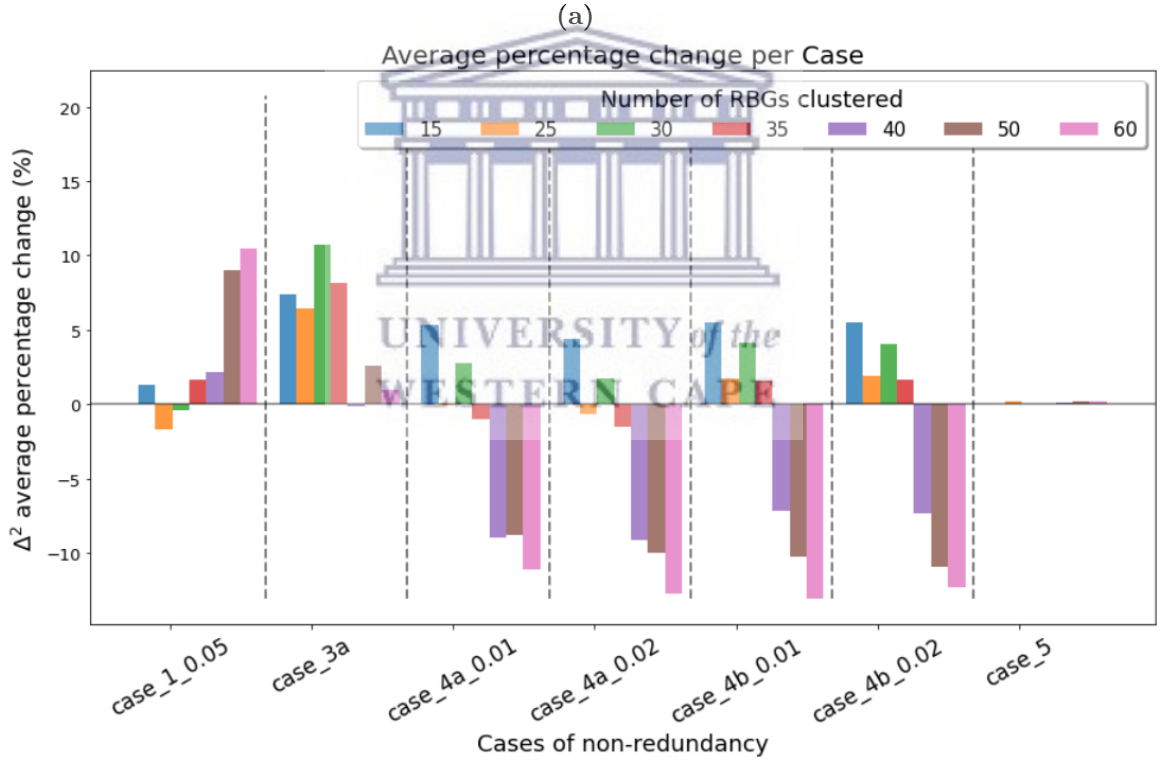


(b)

Figure 5.28: bar graphs showing average results for option2 of modifying redcal. These results show k-values in the range [2, 6]. Plot (a) shows the number of antennas (out of 124) that have improved their gain solution. Plot (b) shows the percentage change calculated using equation 5.6, and any positive value/percentage indicate the improvement of logi_cal and negative percentages show the opposite for each case of non-redundancy.



(a)



(b)

Figure 5.29: bar graphs showing average results for option2 of modifying redcal. These results show k-values in the range [4,6]. Plot (a) shows the number of antennas (out of 124) that have improved their gain solution. Plot (b) shows the percentage change calculated using equation 5.6, and any positive value/percentage indicate the improvement of logi_cal and negative percentages show the opposite for each case of non-redundancy.

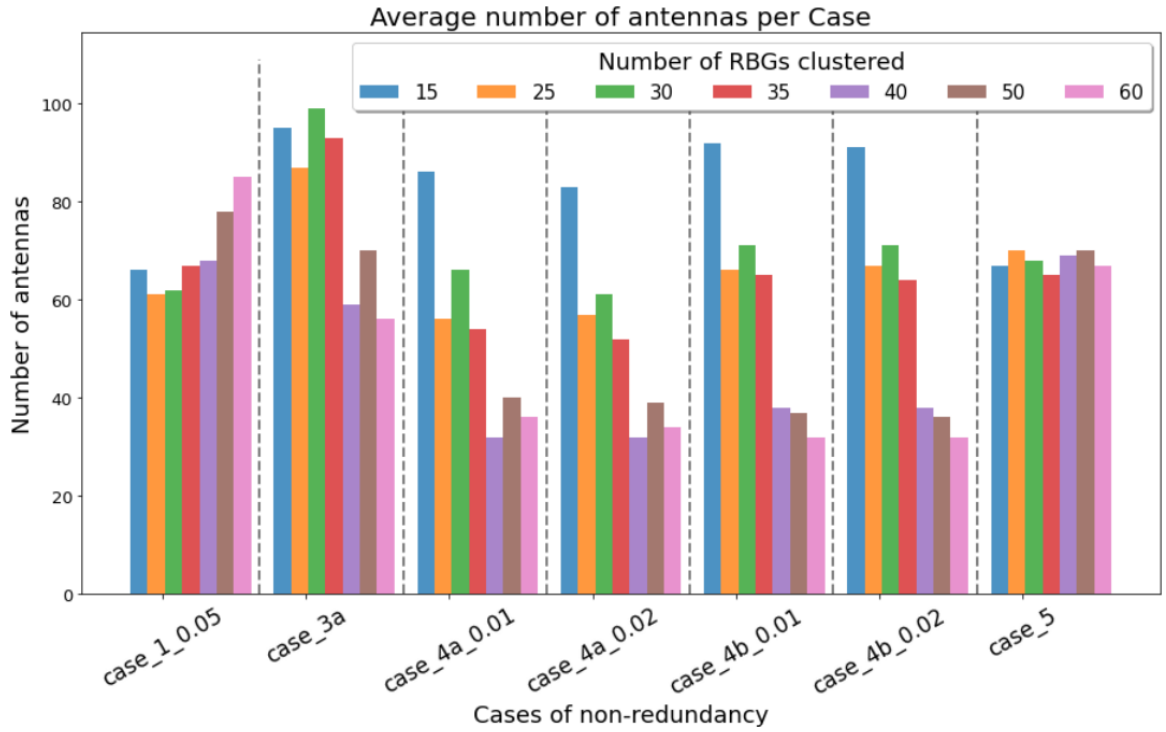
5.5.5 Logi_cal Option 3

For this method involves we cluster the specified RBGs using different k-values at random. This randomness is is limited to a range of k-values staring from k=2 to either k=3,4,5 or 6. With this method its easy to see that the more k-values we have to choose from will result in differing results for every iteration. For example (look at Table .6) when we clustered the first 40 RBGs and k-range=[2,3], then each group will be clustered in either 2 or 3 sub-groups and the margin of error for each iteration is about ±6 antennas. However for k-range=[2,5] the margin of error for each iteration is about ±11 antennas depending on the case of non-redundancy. This shows that this margin of error could be reduced if we limit the range of k-values to be considered in the calibration, to only two choices in k-values (either 2/3, 3/4, 4/5, 5/6 or any combination of two k values).

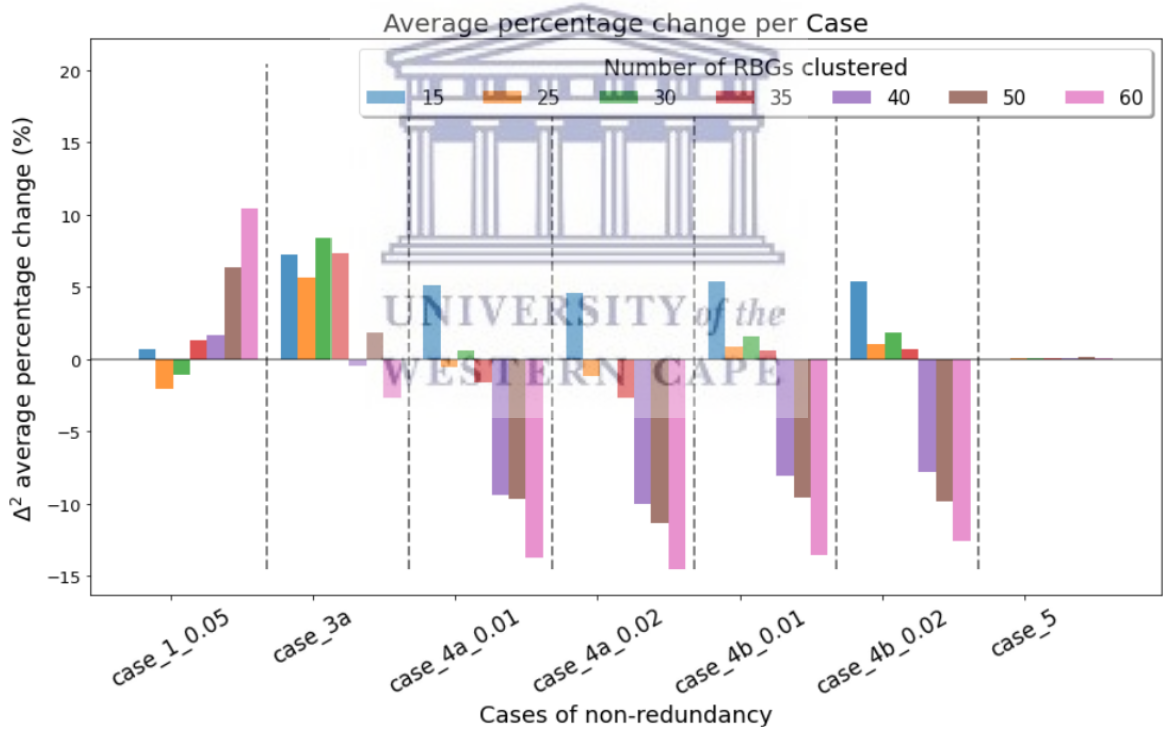
Figure 5.30 and Figure 5.28(a) shows the average number of antennas, out of 124, that have improved their gain solution. From these figures we can see that for all variations of non-redundancy in case 4, the number of antennas that have improved gain solutions has decreased significantly, when more redundant baseline groups are clustered. This can also be seen in Figure 5.30 and Figure 5.28(b), where the overall percentage change in the antenna gain solution seems to get much worse when more RBGs are clustered. This means that increasing the number of RBGs to cluster makes the calibration method worse for case 4 (ellipticity and rotation of the primary beam) non-redundancy. Since the RBGs are arranged in order by length, from short to long, we can deduce from the plot that clustering only baselines with shorter lengths improves the gain solutions for case 4 and the opposite is true for case 1 (sidelobe-only perturbations).

Case 5 (stretching the baseline length) is different from the other case of non-redundancy since it shows neither improvement nor deterioration in the antenna gain solutions, especially when we look at the average percentage change, for Figures 5.28, 5.29 and 5.30(b). This is true for any approach/option tested here for improving the calibration method. For case 3a (stretching the primary beam), when we look at Figure 5.28 and Figure 5.29(b), we can see that any number of RBGs clustered will improve the gain solutions, since every bar is in the positive side of the y-axis. But we do not see the same trend for Figure 5.30(b), where the gain solutions deteriorate when we cluster more baselines. This means that on average, the

calibration approach of using option2 works much better than option3 in improving the gain solutions.



(a)



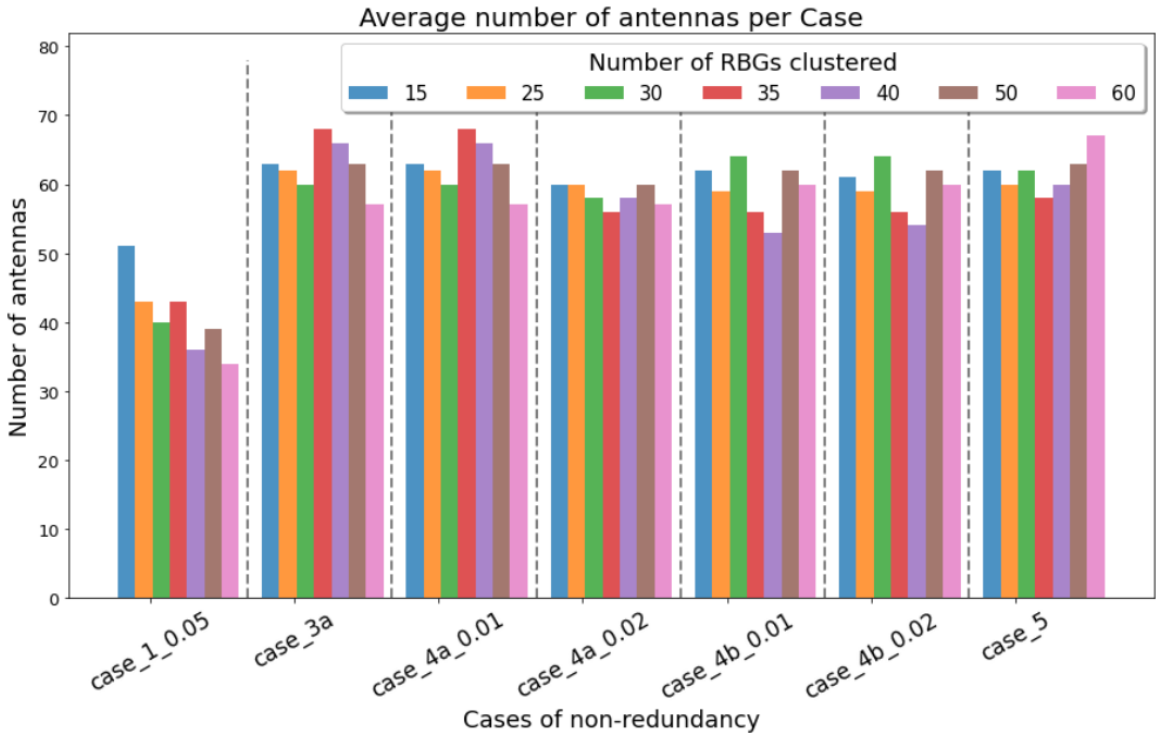
(b)

Figure 5.30: bar graphs showing average results for option3 of modifying redcal, using k-values up to the range [2,6]. Plot (a) shows the number of antennas (out of 124) that have improved their gain solution. From plot (b) any positive value/percentage indicate the improvement of logi_cal and negative percentages show the opposite for each case of non-redundancy.

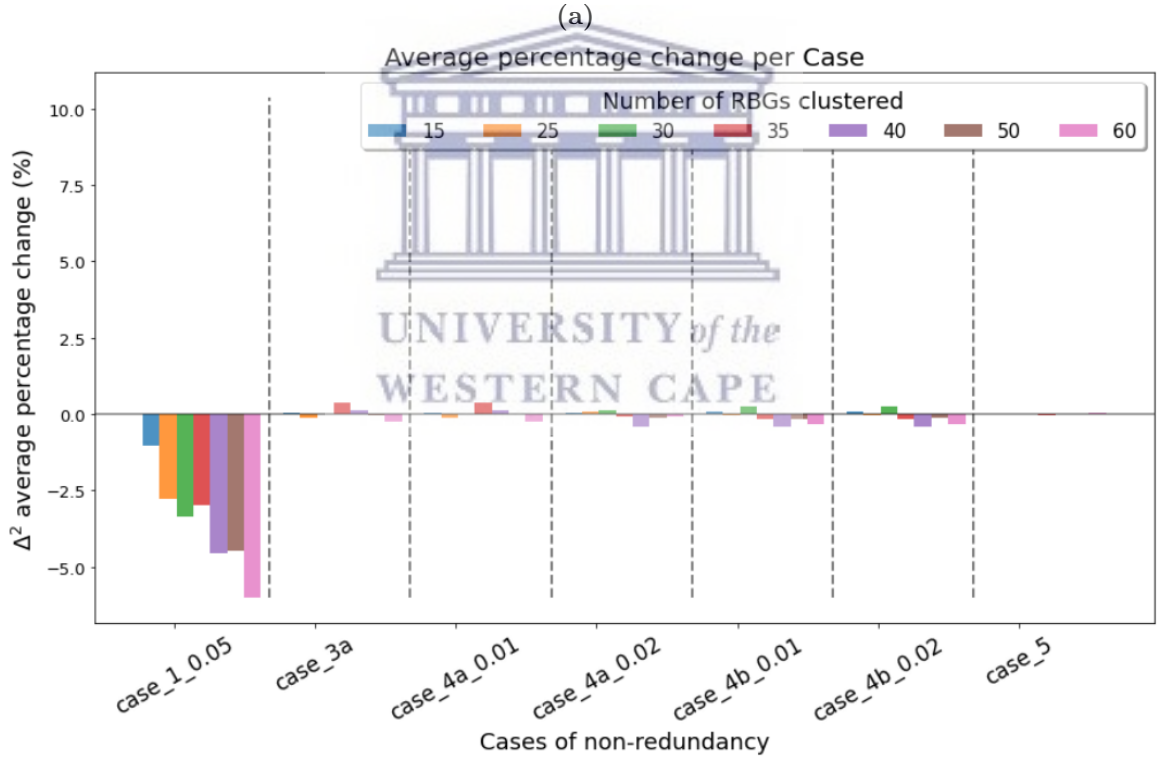
5.5.6 Logi_cal Option 4

This method is very different from the other three methods, because this does not include a clustering algorithm. In this method we use the same k-value however since there is no clustering algorithm, the baselines are clustered at random. For example if k=2, it means we want to cluster the baselines within a redundant baseline group into two groups, so each baseline will randomly be given a label of 0 or 1. Then after the baselines have been given a label, those with the same label will be clustered into the same group. This method is mainly introduced to try and validate the need for a clustering algorithm. This is to compare if any improvement observed using this calibration method (logi_cal) is due to the clustering algorithm or it is due to the simple action of randomly splitting the RBGs.

What we observe from Figure 5.31(b) is that we see no improvement in the gain solutions, in fact, especially for case_1_0.05 the gain solutions are getting much worse. When we compare this result to the other three options highlighted (which all use a clustering algorithm), we can see that there is a need to use a clustering algorithm to improve the gain solutions. So to summarise it shows we need a clustering algorithm in our method, to have a machine distinguish between baselines and group those with similar traits into clusters for calibration.



(a)



(b)

Figure 5.31: bar graphs showing average results for option3 of modifying redcal, using k-values up to the range [2,6]. Plot (a) shows the number of antennas (out of 124) that have improved their gain solution. From plot (b) any positive value/percentage indicate the improvement of logi_cal and negative percentages show the opposite, for each case of non-redundancy.

5.5.7 Optimizing hyperparameters

In this sub-section we try to optimize the hyperparameters used during calibration: The number of RBGs and the number of clusters we want. We do this by testing which hyperparameter is most influential in improving the gain solutions. We ran a simulation where we know the perturbation levels for each of the 124 antennas, making it easier to understand which antenna pair make the most redundant baseline group. By understanding the data, we can determine the number of clusters that we want to divide the data points into. This means we are only left with the number of RBGs to cluster as an unknown. Using this dataset we can also figure out the influence the k-value has on the overall performance of the logi_cal.

Below are the perturbation levels on the primary beam added to the simulation for case 3a, with nfreq=120 and ntimes=10 .

```
1 xstretch = 0*xstretch + 1
2 xstretch[Nant//2:] *=1.02
3 ystretch = 0*ystretch + 1
4 rotation = 0*rotation
5 mainlobe_scale= 0*mainlobe_scale + 1
```

Instead of the perturbation levels being random, in *line 2*: The first half of the array(antennas) are all ones and the second half are 1.02; *line 3*: all the antennas have a ystretch values of 1; *line 4*: all the antennas have a rotation values of 0; *line 5*: all the antennas have a mainlobe_scale values of 1.

With no knowledge of k in K-means to use during calibration, it helps to have some prior knowledge of the dataset. This knowledge will be invaluable in making sure the results of the K-means clustering algorithm make sense given the context of the problem. This is done by simulating the array with known perturbation levels (A and B), A=1, B=1.02. Therefore for our redundant baseline groups, we could only have three combinations AA, BB and AB. A better representation of these combinations is shown in Figure 5.32, where we could only have baselines made out of antenna combinations: blue-blue; blue-red and red-red. So the hope is to see if the clustering algorithm can be able to split them since we already know which antenna pairs are redundant based on their perturbation levels.

So since we now know precisely which baselines form the most redundant baseline

group, we can use that to test the performance of the clustering algorithm. We want to know whether all the baselines in the subgroups follow the predicted cluster allocations (AA, BB and AB). To answer this, we wrote a script that will manually cluster the baselines based on the perturbation levels of each antenna and then compare it to the output of the clustering algorithm. These results are shown in Figure 5.33. This result is very significant because it proves two things: (1) The chosen summary statistic works (using all-time samples and frequency channels); (2) The clustering algorithms works on visibility solutions, whether the system has added noise or gains.

So with the perturbations not randomized like the previous simulations, we can now try to infer the k-value from the results we get after running redcal and logi_cal. The idea is to see if we could use the perturbation levels to automatically get the k-value every time we run logical instead of using a range of k-values to find the one that gives us the best performance.

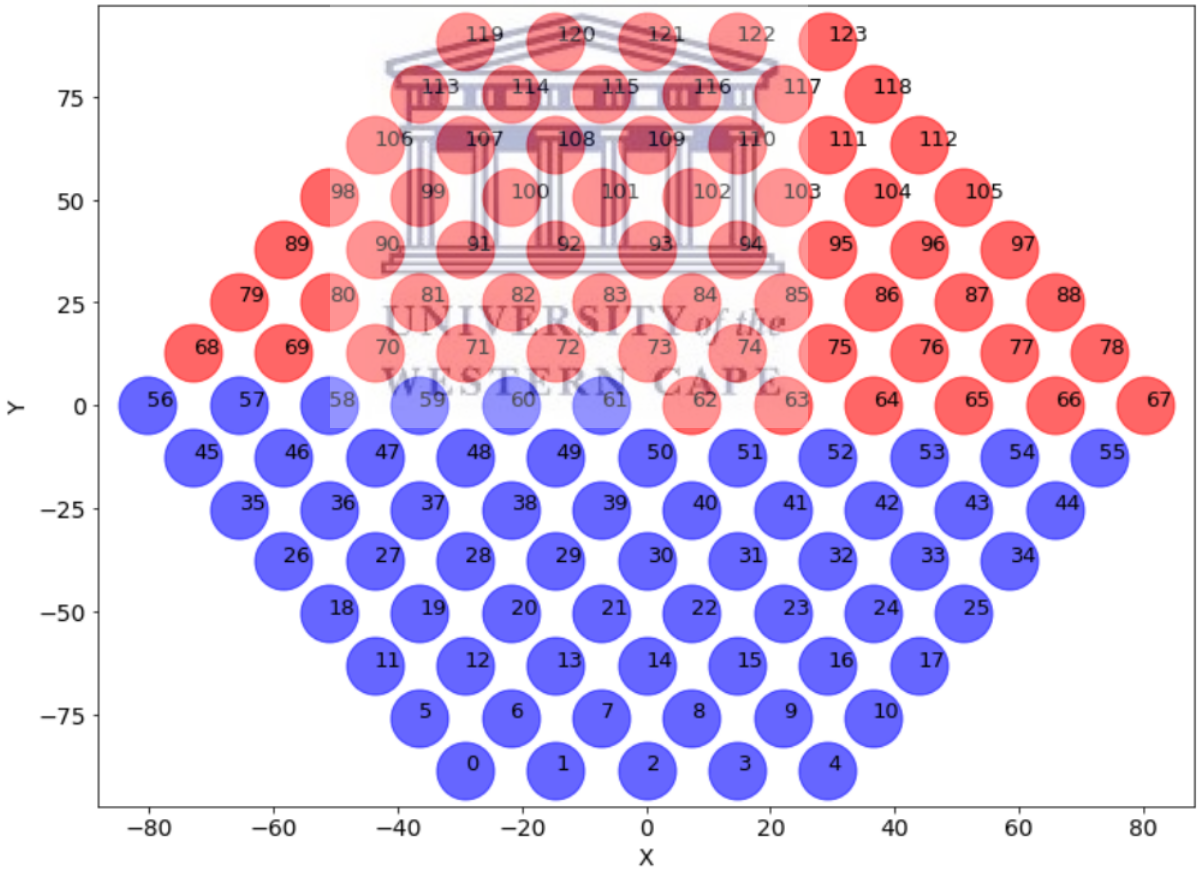
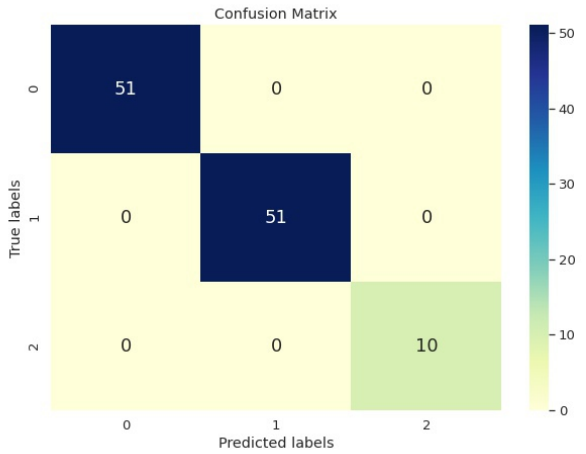
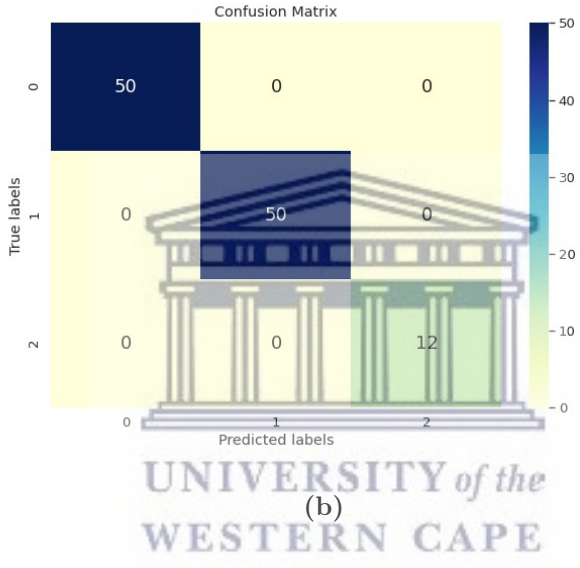


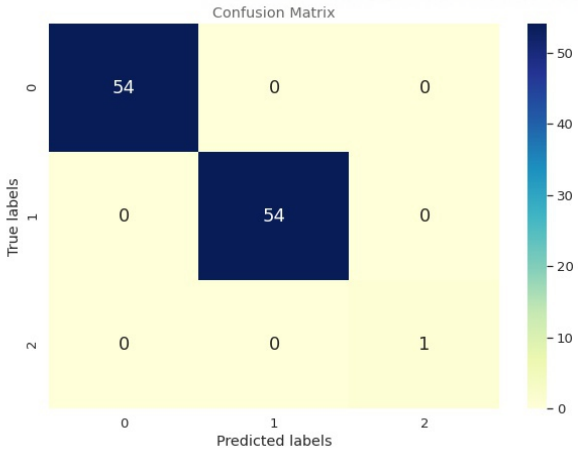
Figure 5.32: This is an image of 124 antenna array, showing the position of the antennas separated by meters represented by the numbers in the x-y axes. The image shows antennas in blue with A=xstretch=1 and antennas in red with B=xstretch=1.02



(a)



(b)



(c)

Figure 5.33: Confusion matrices for the first 3 redundant baseline groups with baseline length of 14.6-m and orientation of (a) 120°, (b) 60° and (c) 0° to the Y-axis. The angle is calculated counter clockwise to the East. All the confusion matrices show a 100% accuracy and precision when using the clustering algorithm on the true data and the data with noise and gains.

K-Means(true_data) Vs clusters based on perturbation

After clustering the baselines based on what we know about the perturbation levels, we get a 100% percent accuracy on all the 60 RBGs clustered. This is really promising, it shows that using the visibility solutions as inputs for the clustering algorithm works as a way to identify redundancy between baselines. Not only does this validate the visibility solutions, it also shows that k-means does what it is meant to do.

K-Means (true_data) Vs K-Means (noise_gains_data)

Since K-Means(true_data) and clusters based on perturbation are basically indistinguishable we could use either to compare with data with noise and gains. Except for RBG6 and RBG54, all the RBGs give us a 100% accuracy when clustering. This clearly shows that even with added noise and gains k-means can still recover the same groups as when using true data. This also goes with the fact that we already know the number of sub-groups (the k-value) we can get given that we know the perturbation levels. So to get this 100% accuracy is only possible because we know the ideal k-value.

Now what happens in a real-world case where we do not know the individual perturbation for each antenna, thereby not knowing the ideal k-value for clustering. The recovery of the same clusters is highly dependent on the perturbation levels added to the system, because when clustering data with random perturbations we cannot recover a 100% of the subgroups/clusters.

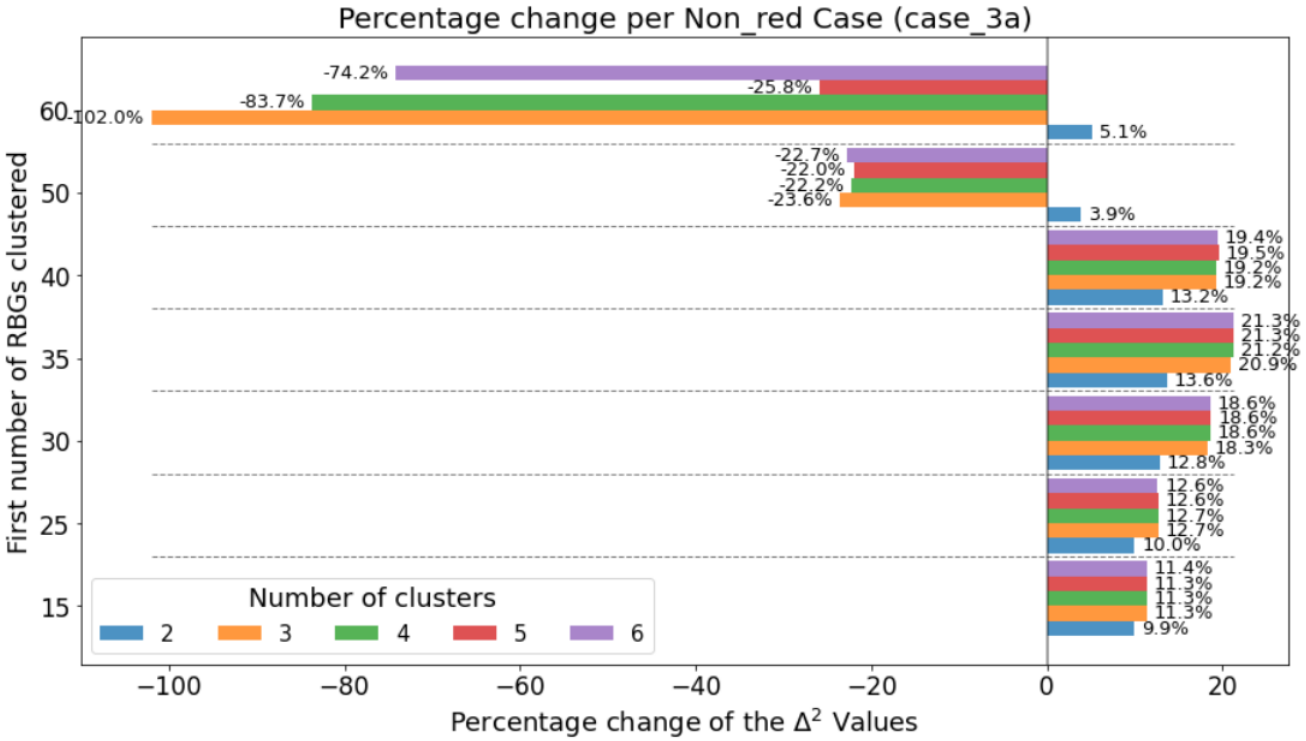


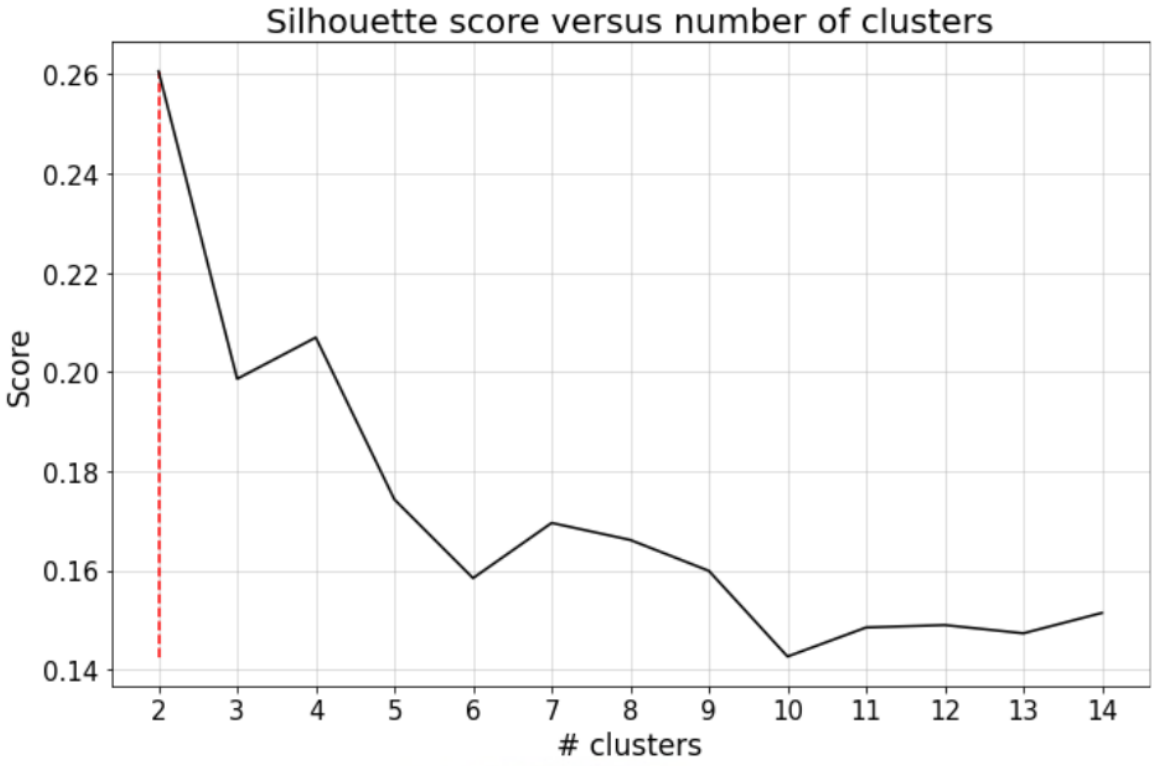
Figure 5.34: Plot represents the percentage change of the Δ^2 values (positive values indicate that logi_cal has improved the overall gain solutions by a certain percentage). On the y-axis we have the first number of redundant baseline groups that are clustered in to more redundant subgroups. This plot is for case3a where we know the perturbation levels. This shows that for the most part the k-values are giving us almost the same percentage change for a given number of RBGs used.

The results in Figure 5.34 indicate that the k-value used for clustering does not matter that much compared with the number of redundant baseline groups chosen to be clustered. This is very evident as there is some change in the percentage changes for any number of RBGs clustered, and on the other hand, it does not change that much when using k=3,4,5,6. This also tells us that clustering the RBGs into two groups (k=2) can improve the calibration method regardless of the number of RBGs clustered. We also see that using k=2 for calibration improves the gain solutions no matter the approaches (or options) tested to modify redcal. However, in most cases, this improvement is not that desirable compared to when using other k-values. Since this is a simulation, we have the luxury to know what the optimal k-value is, which is something we do not have in a real-life situation. In summary, k=2 is guaranteed to improve redundant calibration for simulated data (but not by that much) and the number of RBGs to cluster is the most important parameter to consider when

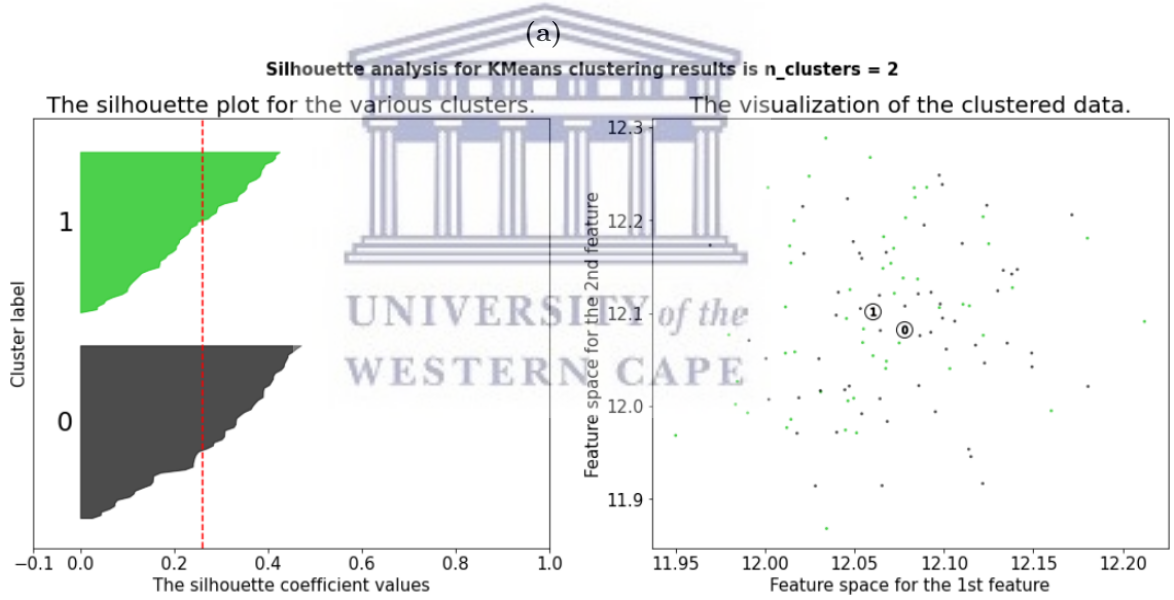
running logi_cal.

Even though we have shown that the k-value does not have that much of an effect on the overall performance of the calibration method, it could still be useful to find a method that can automatically identify the optimal k-value. We discussed two methods in section 3.2.1 that are very popular in finding the optimal k-value to use during clustering, the elbow method and the silhouette score. The elbow method is a visual analysis approach, where optimal the k-value is the point where the plot begins to diminish (forming an elbow). As shown in Figure 5.36 finding the diminishing factor in this plot is subjective at best, making it difficult to find the optimal k-value. Since this method is visual, it is hard to automate this in our calibration method. The other method is the silhouette score which is another graphical approach, however, unlike the elbow method, it shows a concise graphical representation of how well each object has been clustered. It also has a clear definition of what makes an optimal k-value. By definition, the k-value with the highest silhouette score is the optimal k-value (2), as shown in Figure 5.35(a).

Figure 5.35(b) shows the distribution of the data for k=2, which is the optimal k-value given by the silhouette score. From this plot you can see that there is no clear structure of a cluster, we can also see it in the low value of the silhouette score, meaning that the confidence of k=2 being the ideal k-value is very small. Since we know how the data is structured, we already know that k=3 is the optimal k-value and both these methods fail to identify that. This means both these methods are not ideal to find the best k-value and will require that calibration be run with different k-values.



(a)



(b)

Figure 5.35: The plots show the optimal k-value to be used to cluster a redundant baseline group for case3a. (a) shows the silhouette score against the number of clusters. (b) shows the distribution of the data for the optimal k-value given by the silhouette score.

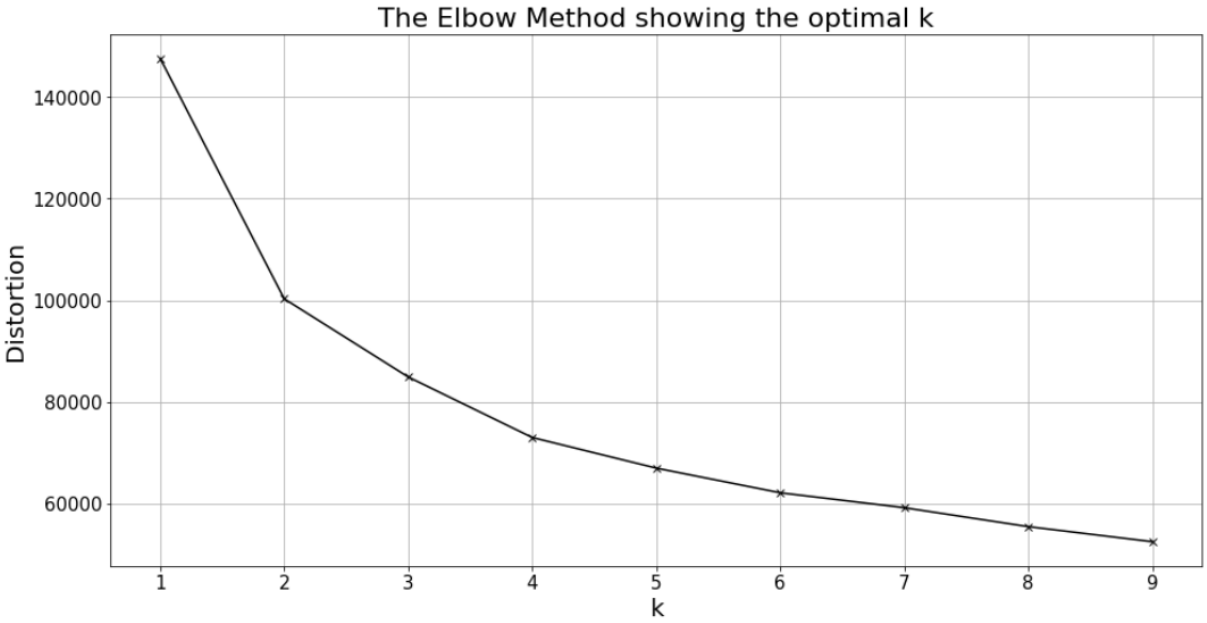


Figure 5.36: This Image shows results the elbow method, for determining the optimal k-value for case 3a. On the x-axis we have the k-value and on the y-axis is the distortion (the average of the squared distances from the cluster centers of the respective clusters).

6 Conclusion

Observing the 21 cm line promises to provide us with a better view of the epoch of reionisation, however separating the 21 cm signal and the astrophysical foregrounds is a challenge. This separation requires precise calibration of the system. Currently, several experiments have been built that have precise control of systematics, with sufficient sensitivity for the detection and characterization of the cosmological 21cm signal. These instruments are designed differently from traditional radio telescopes, and because of the precision they are trying to achieve, they will require non-traditional ways of calibrating them so they could reach their design sensitivities. HERA is such an instrument, which uses the internal consistency of the array to calibrate its data, called redundant baseline calibration(redcal) (Dillon & Parsons, 2016; Dillon et al., 2020). In redcal antenna pairs with the same positional vector between them see the same signal. However, there are many reasons why this assumption is not always true, due to many reasons that can spoil the redundancy of the array like different primary beam patterns as well as 'outlier' antennas which sustain the largest gain errors (Barry et al., 2016; Ewall-Wice et al., 2016; Joseph et al., 2018; Orosz et al., 2019; Gehlot et al., 2021; Choudhuri et al., 2021).

Due to the uncertainty of what might spoil the redundancy of the array, in this thesis, we try to classify the non-redundancy in the HERA array using a machine learning clustering algorithm. So we are clustering the baselines within a redundant baseline group (RBG) based on their visibility solutions. We then use these clustered baselines that are now more redundant to calibrate the data with a new calibration technique called logi_cal. In this technique, we have modified the original redcal code to include a section where the baselines within the redundant baseline groups (predetermined by their positional vector) are clustered into sub-groups based on their visibility solutions. To modify redcal we focused on the primary assumption that baselines within a redundant baseline group should see the same Fourier mode on the sky. So using a machine learning algorithm, we perform the clustering on a single redundant baseline group using the visibility solutions from redundant calibration (specifically, the *hera_cal* package), this is because these visibility solutions are, in principle, the same.

In chapter 4 we described the simulation and the cases of non-redundancy used as our data. We also describe the steps we took in modifying redcal and also

outlined different options/approaches we took modifying redcal. In option 1, all the specified RBGs are clustered using the same k-value; option 2 involves clustering all the specified RBGs using different k-values (uniform distribution of k-values) and option 3 involves clustering all the specified RBGs using random k-values within a given range. Although options 1-3 are completely different, there are no notable distinctions in the results we get. Except when we compare them with option 4 (the baselines are clustered randomly without any use of a clustering algorithm) which is used as a way to validate the use of the clustering algorithm. These results clearly show that just increasing the number of groups will not improve the results. There needs to be a strategical way of grouping those baselines, and using a clustering algorithm is the best way of doing it. The results also show that for any case of non-redundancy and whatever option (1-3) is implemented, k=2 is always improving the results. This improvement might be small compared to other k-values for some cases, but k=2 for the simulations is always guaranteed to improve the antenna gain solutions.

In chapter 5 we start showing and discussing the results, by comparing the results we get from redcal and logi_cal using gain solutions. These results show that logi_cal can improve the results for any beam dependent primary beam perturbations added to the simulation, but that improvement is not extended for positional non-redundancies added to the simulation. Overall per case of non-redundancy: Case3a_0.01 the calibration is improved by 12.2%, case1_0.01 by 13.6% and case4a and case 4b the calibration is improved by about $6.8 \pm 0.1\%$. Unlike the other cases, case5 (stretching the baseline length by 10cm) is the only one that is positional non-redundancy, and our results show that logi_cal does not improve the antenna gains. This could be explained by the fact that when a redundant array like HERA, has positional offsets in the order of 10 cm, careful consideration has to made on when to do redundant calibration (Joseph et al., 2018). In this chapter we also validated the use of a clustering algorithms and the choice of features for the visibility solutions. The way this was done is to simulate a case where we already know the perturbation levels and the baseline redundant subgroups that will be formed as a result, and our method was able to recovered those subgroups, clearly showing that the clustering algorithm is performing as expected.

Despite this success there is still more to be done to make even more improvements, for example this version of logical still requires two parameters before run-

ning calibration (1) the number of redundant baseline groups to be clustered (2) the number of clusters we want (the k in k-means). For (1) the number of RBGs to be clustered varies on a case to case basis, some cases of non-redundancy may require more RBGs to be clustered and some may require less. Because of this result, using $80m$ for the maximum baseline cut length (max_bl_cut) for calibration, gives us a reasonable number of RBGs to cluster. For (2) we could try to find a more automated way of choosing the number of clusters by using algorithms like the elbow method and the silhouette score. We say 'like' here because these methods do not work for our datasets so there might be some other methods out there that could work, as shown in chapter 5.5.7. Another thing that chapter 5.5.7 also showed is that parameter (1) is more important to optimise than (2), since the improvement of the results was more driven by the number of redundant groups clusters and not the k-values. So this leaves more opportunities for improvements to be made on this project.

Bibliography

Aha D. W., Kibler D., Albert M. K., 1991, Machine learning, 6, 37

Anderberg M. R., 1973, Cluster Analysis for Applications. Vol. 19, Academic press

Arbelaitz O., Gurrutxaga I., Muguerza J., Pérez J. M., Perona I., 2013, Pattern Recognition, 46, 243

Arthur D., Vassilvitskii S., 2006, Technical report, k-means++: The advantages of careful seeding. Stanford

Bagla J., Loeb A., 2009, arXiv preprint arXiv:0905.1698

Baron D., 2019, arXiv preprint arXiv:1904.07248

Barry N., Hazelton B., Sullivan I., Morales M., Pober J., 2016, Monthly Notices of the Royal Astronomical Society, 461, 3135

Bernardi G., et al., 2009, Astronomy & Astrophysics, 500, 965

Birsan T., Tiba D., 2005, in IFIP Conference on System Modeling and Optimization. pp 35–39

Bock T., 2020, What is a dendrogram?, https://www.displayr.com/what-is-dendrogram/

Boden A. F., 2007, New Astronomy Reviews, 51, 617

Bottou L., Bengio Y., 1995, in Advances in neural information processing systems. pp 585–592

Bradley P. S., Fayyad U. M., 1998, in ICML. pp 91–99

Braun R., 2013, Astronomy & Astrophysics, 551, A91

Byrne R., et al., 2019, The Astrophysical Journal, 875, 70

Cardie C., 1993, in Proceedings of the tenth international conference on machine learning. pp 25–32

Celebi M. E., Kingravi H. A., Vela P. A., 2013, Expert Systems with Applications, 40, 200

Chandola V., Banerjee A., Kumar V., 2009, ACM computing surveys (CSUR), 41, 1

Choudhuri S., Bull P., Garsden H., 2021, Monthly Notices of the Royal Astronomical Society, 506, 2066

Cisewski J., Croft R. A., Freeman P. E., Genovese C. R., Khandai N., Ozbek M., Wasserman L., 2014, Monthly Notices of the Royal Astronomical Society, 440, 2599

Condon J. J., Ransom S. M., 2016, Essential radio astronomy. Princeton University Press

Datta A., Bowman J., Carilli C., 2010, The Astrophysical Journal, 724, 526

Day C., 2015, Adrian Liu's Summer 2015 Global Signal Talks: Part 2: Evolution of the Global 21cm Signal, https://casper.astro.berkeley.edu/astrobaki/images/9/9f/GlobalSignalTalk2_ALiu_Summer2015.pdf

De Amorim R. C., Hennig C., 2015, Information sciences, 324, 126

De Swart J., Bertone G., van Dongen J., 2017, Nature Astronomy, 1, 1

Deboer D. R., et al., 2017, Publications of the Astronomical Society of the Pacific, 129

Dillon J. S., 2017, Proceedings of the International Astronomical Union, 12, 114

Dillon J. S., Parsons A. R., 2016, The Astrophysical Journal, 826, 181

Dillon J. S., et al., 2018, Monthly Notices of the Royal Astronomical Society, 477, 5670

Dillon J. S., et al., 2020, Monthly Notices of the Royal Astronomical Society, 499, 5840

Eiter T., Mannila H., 1994, Technical report, Computing discrete Fréchet distance. Citeseer

Ewall-Wice A., et al., 2016, The Astrophysical Journal, 831, 196

Ewall-Wice A., Dillon J. S., Liu A., Hewitt J., 2017, Monthly Notices of the Royal Astronomical Society, 470, 1849

Ewen H. I., Purcell E. M., 1951, Nature, 168, 356

Fernández A., Gómez S., 2008, Journal of Classification, 25, 43

Fialkov A., Loeb A., 2013, Journal of Cosmology and Astroparticle Physics, 2013, 066

Forgy E. W., 1965, biometrics, 21, 768

Fréchet M., 1906, Rendiconti del Circolo Matematico di Palermo (1884-1940), 22, 1

Furlanetto S. R., 2016, in , Understanding the Epoch of Cosmic Reionization. Springer, pp 247–280

Furlanetto S. R., Oh S. P., Briggs F. H., 2006, Physics reports, 433, 181

Gary D. E., 2019, PHYSICS 728, RADIO ASTRONOMY, https://web.njit.edu/~gary/728/

Gehlot B. K., et al., 2021, Monthly Notices of the Royal Astronomical Society, 506, 4578

Gonzalez T. F., 1985, Theoretical computer science, 38, 293

Goodfellow I., Bengio Y., Courville A., 2016a, Deep learning. MIT press

Goodfellow I., Bengio Y., Courville A., 2016b, Deep learning, 1, 98

Gorthi D. B., Parsons A. R., Dillon J. S., 2021, Monthly Notices of the Royal Astronomical Society, 500, 66

Grobler T. L., Bernardi G., Kenyon J. S., Parsons A. R., Smirnov O. M., 2018, Monthly Notices of the Royal Astronomical Society, 476, 2410

Haarlem M., et al., 2013, Astronomy and Astrophysics

Harker G., et al., 2010, Monthly Notices of the Royal Astronomical Society, 405, 2492

Hossain M. Z., Akhtar M. N., Ahmad R. B., Rahman M., 2019, Indonesian Journal of Electrical engineering and computer science, 13, 521

Huang Z., 1998, Data mining and knowledge discovery, 2, 283

Huang J., Chai J., Cho S., 2020, Frontiers of Business Research in China, 14, 1

Hurley-Walker N., et al., 2017, Monthly Notices of the Royal Astronomical Society, 464, 1146

Hurley-Walker N., et al., 2019, Publications of the Astronomical Society of Australia, 36

Jacobs D. C., et al., 2016, The Astrophysical Journal, 825, 114

Jain A. K., Murty M. N., Flynn P. J., 1999, ACM Comput. Surv., 31, 264–323

Jansky K. G., 1932, Proceedings of the Institute of Radio Engineers, 20, 1920

Jansky K. G., 1933, Nature, 132, 66

Jarvis M. J., Bacon D., Blake C., Brown M. L., Lindsay S. N., Raccanelli A., Santos M., Schwarz D., 2015, arXiv preprint arXiv:1501.03825

Jordan M. I., Mitchell T. M., 2015, Science, 349, 255

Joseph R. C., Trott C. M., Wayth R. B., 2018, The Bias and Uncertainty of Redundant and Sky-based Calibration Under Realistic Sky and Telescope Conditions (arXiv:1810.11237), doi:10.3847/1538-3881/aaec0b, http://arxiv.org/abs/1810.11237http://dx.doi.org/10.3847/1538-3881/aaec0b

Kanekar N., Chengalur J., Lane W., 2007, Monthly Notices of the Royal Astronomical Society, 375, 1528

Kern N. S., et al., 2020, Absolute Calibration Strategies for the Hydrogen Epoch of Reionization Array and Their Impact on the 21 cm Power Spectrum (arXiv:1910.12943), doi:10.3847/1538-4357/ab67bc, https://arxiv.org/abs/1910.12943

Knauer C., Löffler M., Scherfenberg M., Wolle T., 2011, Theoretical Computer Science, 412, 4173

Kodinariya T. M., Makwana P. R., 2013, International Journal, 1, 90

Koopmans L., et al., 2015, arXiv preprint arXiv:1505.07568

Kwon D., 2017, What ended the dark ages of the universe?, https://sciencesprings.wordpress.com/2017/02/07/from-symmetry-what-ended-the-dark-ages-of-the-universe/

La Plante P., et al., 2021, Astronomy and Computing, 36, 100489

Le Cessie S., Van Houwelingen J. C., 1992, Journal of the Royal Statistical Society: Series C (Applied Statistics), 41, 191

LeCun Y., Bengio Y., Hinton G., 2015, nature, 521, 436

Lian X., Xu H., Zhu Z., Hu D., 2020, Monthly Notices of the Royal Astronomical Society, 496, 1232

Litjens G., et al., 2017, Medical image analysis, 42, 60

Liu A., Tegmark M., 2011, Phys. Rev. D, 83, 103006

Liu A., Tegmark M., Morrison S., Lutomirski A., Zaldarriaga M., 2010, Monthly Notices of the Royal Astronomical Society, 408, 1029

Lloyd S., 1982, IEEE transactions on information theory, 28, 129

Lonsdale C. J., et al., 2009, Proceedings of the IEEE, 97, 1497

MacQueen J., et al., 1967, in Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. pp 281–297

Madau P., Meiksin A., Rees M. J., 1997, The Astrophysical Journal, 475, 429

Mahmood A. N., Hu J., Tari Z., Leckie C., 2010, Journal of Network and Computer Applications, 33, 491

Mahmoud M., Ensor A., Biem A., Elmegreen B., Gulyaev S., 2011, Studies in Computational Intelligence, 426

Marom Y., Feldman D., 2019, CoRR, abs/1903.06904

Marr J. M., Snell R. L., Kurtz S. E., 2015, Fundamentals of radio astronomy: observational methods. CRC Press

Medar R., Rajpurohit V., Rashmi B., 2017. pp 1–6, doi:10.1109/ICCUBEA.2017.8463779

Michelson A. A., 1890, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 30, 1

Michelson A. A., 1920, The Astrophysical Journal, 51, 257

Michelson A. A., Pease F. G., 1921, Proceedings of the National Academy of Sciences of the United States of America, 7, 143

Mitchell D. A., Greenhill L. J., Wayth R. B., Sault R. J., Lonsdale C. J., Cappallo R. J., Morales M. F., Ord S. M., 2008, IEEE Journal of Selected Topics in Signal Processing, 2, 707

Morales M. F., Hewitt J., 2004, The Astrophysical Journal, 615, 7

Müller M., 2007, Information retrieval for music and motion, pp 69–84

Murphy K. P., 2012, Machine learning: a probabilistic perspective. MIT press

Murtagh F., 1984, Computational Statistics Quarterly, 1, 101

Murtagh F., Contreras P., 2012, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2, 86

Natarajan A., Yoshida N., 2014, Progress of Theoretical and Experimental Physics, 2014, 06B112

Nikolic B., Carilli C., 2017, in 2017 32nd General Assembly and Scientific Symposium of the International Union of Radio Science, URSI GASS 2017. pp 1–4 (arXiv:1709.05245), doi:10.23919/URSIGASS.2017.8105387, https://arxiv.org/abs/1709.05245

Noordermeer E., Van Der Hulst J., Sancisi R., Swaters R., Van Albada T., 2005, Astronomy & Astrophysics, 442, 137

Olson C. F., 1995, Parallel computing, 21, 1313

Orosz N., Dillon J. S., Ewall-Wice A., Parsons A. R., Thyagarajan N., 2019, Monthly Notices of the Royal Astronomical Society, 487, 537

PAPER 2012, Paper, http://eor.berkeley.edu/

Parsons A. R., et al., 2010, The Astronomical Journal, 139, 1468

Parsons A. R., Pober J. C., Aguirre J. E., Carilli C. L., Jacobs D. C., Moore D. F., 2012, The Astrophysical Journal, 756, 165

Penzias A. A., Wilson R. W., 1965, The Astrophysical Journal, 142, 419

Pollard K. S., Van Der Laan M. J., 2002, U.C. Berkeley Division of Biostatistics Working Paper Series, 107

Prestage R. M., Constantikes K. T., Hunter T. R., King L. J., Lacasse R. J., Lockman F. J., Norrod R. D., 2009, Proceedings of the IEEE, 97, 1382

Pritchard J. R., Loeb A., 2012, Reports on Progress in Physics, 75, 086901

Quinlan J. R., Cameron-Jones R. M., 1993, in European conference on machine learning. pp 1–20

Raju N. S., 1988, Psychometrika, 53, 495

Rousseeuw P. J., 1987, Journal of computational and applied mathematics, 20, 53

Santos M. G., Cooray A., Knox L., 2005, The Astrophysical Journal, 625, 575

Selim S. Z., Ismail M. A., 1984, IEEE Transactions on pattern analysis and machine intelligence, pp 81–87

Shaver P., Windhorst R., Madau P., De Bruyn A., 1999, arXiv preprint astro-ph/9901320

Sigurdson K., Furlanetto S. R., 2006, Physical review letters, 97 9, 091301

Singh K., Malik D., Sharma N., 2011, International Journal of Computational Engineering & Management, 12, 105

Smoot G. F., 1998, arXiv preprint astro-ph/9801121

Staveley-Smith L., et al., 1996, Publications of the Astronomical Society of Australia, 13, 243

Su T., Dy J. G., 2007, Intelligent Data Analysis, 11, 319

Sunyaev R., 1974, in Symposium-International Astronomical Union. pp 167–173

Thomas R. M., et al., 2009, MNRAS, 393, 32

Thompson A. R., Moran J. M., Swenson G. W., 2017, Interferometry and synthesis in radio astronomy. Springer Nature

Tiltec 2021, Hydrogen line, https://en.wikipedia.org/wiki/Hydrogen_line#/media/File:Hydrogen-SpinFlip.svg

Trimble V., 1987, Annual review of astronomy and astrophysics, 25, 425

Wang X., Tegmark M., Santos M. G., Knox L., 2006, The Astrophysical Journal, 650, 529

Wijnholds S., Willis A., Salvini S., 2018, Monthly Notices of the Royal Astronomical Society, 476, 2029

Windarto A. P., 2017, International Journal of artificial intelligence research, 1, 26

Witowski K., Stander N., 2012. , doi:10.2514/6.2012-5580

Wu J., 2012, Advances in K-means clustering: a data mining thinking. Springer Science & Business Media

Xiong L., Póczos B., Schneider J., Connolly A., VanderPlas J., 2011, in Proceedings of the fourteenth international conference on artificial intelligence and statistics. pp 789–797

Xu R., Wunsch D., 2005, IEEE Transactions on neural networks, 16, 645

Zaldarriaga M., Furlanetto S. R., Hernquist L., 2004, The Astrophysical Journal, 608, 622

Zaroubi S., 2010, Probing the Epoch of Reionization with Low Frequency Arrays (arXiv:1002.2667)

Zaroubi S., 2013, The First Galaxies, pp 45–101

Zheng H., et al., 2014, Monthly Notices of the Royal Astronomical Society, 445, 1084

automaticaddison a., 2019, K-means clustering and the local search problem, https://automaticaddison.com/k-means-clustering-and-the-local-search-problem/

de Oliveira-Costa A., Tegmark M., Gaensler B., Jonas J., Landecker T., Reich P., 2008, Monthly Notices of the Royal Astronomical Society, 388, 247

van Haarlem M. P., et al., 2013, Astronomy & astrophysics, 556, A2

van de Hulst H. C., 1945, Nederlandsch Tijdschrift voor Natuurkunde, 11, 210

Appendix A:
Additional information on the results presented

.1

N_antennas = 75, nfreq=120, ntimes=10, max_bl_cut = 80						
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)
Case 3a_0.01	2	56	54	56	58	45
	3	50	47	53	50	34
	4	55	53	60	62	43
	5	58	52	62	62	38
	6	55	51	60	61	38
	Avg	55	51	58	59	40
Case 4a_0.01	2	47	31	34	29	22
	3	51	27	32	26	19
	4	58	32	41	29	15
	5	59	34	41	31	18
	6	62	39	42	39	15
	Avg	55	33	38	31	18
Case 4b_0.01	2	48	38	42	38	31
	3	59	35	40	35	26
	4	61	44	51	43	30
	5	60	49	55	50	32
	6	55	53	55	50	32
	Avg	57	44	49	43	30
Case 5	2	36	37	36	37	37
	3	41	38	35	38	37
	4	32	41	40	37	37
	5	40	45	46	45	46

Continued on next page

Table .1 – continued from previous page

N_antennas = 75, nfreq=120, ntimes=10, max_bl_cut = 80						
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)
	6	44	47	49	42	47
	Avg	39	42	41	40	41

Table .1: The table shows a number of antennas that have improved gains out of 75. The numbers on the far left are the k-values used in the clustering algorithm and the numbers on top are the first RBGs clustered, including the longest baseline length present in that group.

.2 Logi_cal Option2

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	2	63	59	61	68	66	82	79
	3	59	59	63	68	73	83	87
	4	62	57	66	57	64	79	79
	5	62	61	63	63	68	82	82
	6	61	65	60	62	64	73	82
	Avg	61	60	63	64	67	80	82
Case 3a	2	93	83	93	87	63	75	71
	3	95	84	100	88	59	72	75
	4	96	89	96	92	75	78	63
	5	94	83	102	96	74	76	74
	6	96	84	98	95	73	66	66
	Avg	95	85	98	92	69	73	70

Continued on next page

Table .2 – continued from previous page

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 4a_0.01	2	87	50	53	43	25	36	31
	3	83	52	58	42	25	36	36
	4	79	50	56	48	29	44	37
	5	87	55	75	50	39	40	39
	6	93	55	71	61	38	37	39
	Avg	86	52	63	49	31	39	36
Case 4a_0.02	2	84	50	50	37	26	34	33
	3	82	50	57	39	26	32	30
	4	80	44	47	42	28	35	36
	5	83	54	65	51	31	43	40
	6	89	57	68	59	40	42	38
	Avg	84	51	57	46	30	37	35
Case 4b_0.01	2	90	54	65	49	32	25	26
	3	78	60	73	49	29	34	32
	4	87	69	64	61	44	46	35
	5	84	60	73	61	41	38	36
	6	93	65	78	77	39	45	34
	Avg	86	62	71	59	37	38	33
Case 4b_0.02	2	94	57	67	53	35	26	28
	3	83	61	66	51	27	37	34
	4	89	65	65	55	42	43	40
	5	93	68	68	58	42	36	35
	6	92	67	80	67	44	45	33
	Avg	90	64	69	57	38	37	34

Continued on next page

Table .2 – continued from previous page

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)

Table .2: The table shows the overall number of antennas with gain solutions when using option 2 of logical to calibrate the data. On this table the k-value starts from k=2 to the specified k-value on the left side of the column.

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	2	0.48	-1.43	-0.1	0.87	0.88	5.41	6.35
	3	0.0	-2.73	-1.03	0.66	3.19	8.57	10.2
	4	0.93	-2.8	-1.02	-1.13	2.11	7.56	8.26
	5	0.41	-1.95	-0.69	0.08	2.27	8.26	7.9
	6	0.27	-1.17	-0.4	0.17	-0.37	4.96	9.93
	Avg	0.42	-2.02	-0.65	0.13	1.62	6.95	8.53
case 3a	2	5.51	4.32	6.93	5.05	0.29	2.32	1.92
	3	5.23	6.31	8.86	4.44	-1.07	2.42	3.31
	4	6.45	5.56	7.96	6.84	2.35	4.4	0.77
	5	6.13	4.8	9.19	7.08	2.46	3.51	3.61
	6	7.69	5.62	8.95	8.53	2.53	2.27	0.04
	Avg	6.2	5.32	8.38	6.39	1.31	2.98	1.93
Case 4a_0.01	2	3.62	-1.8	-0.37	-3.31	-8.88	-9.91	-11.85
	3	3.04	-0.85	0.64	-4.7	-10.88	-12.05	-14.02
	4	3.38	-1.6	-0.69	-3.74	-9.28	-10.31	-12.51
	5	3.71	-1.64	1.32	-1.67	-6.83	-9.11	-13.14
	6	5.8	-1.2	1.01	-0.14	-7.51	-10.7	-12.5
	Avg	3.91	-1.42	0.38	-2.71	-8.68	-10.42	-12.8
Case 4a_0.02	2	3.21	-2.61	-0.92	-4.63	-9.44	-11.04	-12.38
	3	3.06	-1.81	-0.44	-5.52	-12.02	-13.65	-15.13
	4	3.31	-3.2	-2.38	-4.58	-10.6	-12.49	-14.72
	5	3.29	-1.84	0.43	-3.51	-8.28	-10.76	-13.92
	6	5.03	-2.1	1.14	-0.58	-7.72	-12.12	-13.92
	Avg	3.58	-2.31	-0.43	-3.76	-9.61	-12.01	-14.01
Case 4b_0.01	2	3.53	-1.46	-0.23	-2.61	-9.18	-9.68	-11.22
	3	3.0	0.09	1.93	-2.08	-9.42	-8.27	-9.25
	4	4.14	0.19	0.76	-0.86	-6.33	-6.93	-11.88
	5	3.37	-0.74	2.21	0.06	-6.58	-8.79	-12.23
	6	5.43	-0.22	2.86	1.24	-6.08	-7.43	-15.17
	Avg	3.89	-0.43	1.51	-0.85	-7.52	-8.22	-11.95
Case 4b_0.02	2	3.86	-1.32	0.33	-2.08	-8.47	-10.72	-10.0
	3	3.34	0.14	1.55	-2.11	-9.37	-8.33	-8.4
	4	4.34	0.25	0.83	-0.91	-6.33	-6.8	-10.03
	5	3.71	-0.14	1.86	-0.28	-6.53	-8.39	-10.77
	6	5.36	0.46	2.61	0.54	-5.28	-8.01	-10.7
	Avg	4.12	-0.12	1.44	-0.97	-7.2	-8.45	-9.98

Table .3: The table shows the overall percentage change for the Δ^2 values of antenna gain solutions calculated using equation 5.6, and it is the complement of Table .2

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	4	59	61	66	72	72	80	82
	5	69	65	59	66	70	84	85
	6	62	61	61	61	60	77	87
	Avg	63	62	62	66	67	80	85
Case 3a	4	91	93	104	95	58	72	73
	5	97	92	105	98	61	73	61
	6	96	91	104	100	59	67	66
	Avg	95	92	104	98	59	71	67
Case 4a_0.01	4	82	56	72	49	27	41	37
	5	96	68	81	60	34	42	33
	6	90	66	80	61	30	44	35
	Avg	89	63	78	57	30	42	35
Case 4a_0.02	4	76	52	58	47	29	44	36
	5	87	62	81	64	43	45	36
	6	82	63	77	56	38	36	40
	Avg	82	59	72	56	37	42	37
Case 4b_0.01	4	96	70	83	69	44	37	43
	5	89	68	79	63	50	41	32
	6	91	76	87	79	45	34	36
	Avg	92	71	83	70	46	37	37
Case 4b_0.02	4	96	74	81	68	40	40	41
	5	93	71	73	67	46	38	34
	6	91	77	83	74	45	36	35
	Avg	93	74	79	70	44	38	37

Table .4: The table shows the overall number of antennas with gain solutions when using option 2 of logical to calibrate the data. On this table the k-value starts from k=3 to the specified k-value on the left side of the column.

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	4	1.08	-1.78	0.7	2.25	3.93	8.01	8.11
	5	1.93	-1.23	-1.09	2.45	2.7	10.31	10.5
	6	0.92	-2.03	-0.89	0.35	-0.25	8.74	12.81
	Avg	1.31	-1.68	-0.43	1.68	2.13	9.02	10.47
Case 3a	4	6.9	6.85	10.53	7.51	-0.66	2.61	3.36
	5	7.66	6.27	10.6	8.65	0.37	3.47	-1.34
	6	7.56	6.1	11.09	8.21	-0.08	1.71	0.74
	Avg	7.37	6.41	10.74	8.12	-0.12	2.6	0.92
Case 4a_0.01	4	4.81	-1.0	1.53	-2.0	-9.65	-8.41	-10.28
	5	5.61	0.33	3.19	-0.19	-8.23	-8.64	-11.94
	6	5.46	0.38	3.5	-0.71	-8.94	-9.24	-11.24
	Avg	5.29	-0.1	2.74	-0.97	-8.94	-8.76	-11.15
Case 4a_0.02	4	3.78	-1.42	0.33	-3.06	-10.21	-9.4	-12.42
	5	4.48	-0.38	1.83	-0.45	-8.72	-9.5	-13.27
	6	4.87	-0.2	3.11	-1.16	-8.5	-11.2	-12.6
	Avg	4.38	-0.67	1.76	-1.56	-9.14	-10.03	-12.76
Case 4b_0.01	4	5.41	2.43	4.66	1.01	-7.6	-9.7	-11.04
	5	5.24	1.35	3.81	1.24	-6.75	-9.9	-15.18
	6	5.86	1.31	4.02	2.53	-7.17	-11.22	-13.12
	Avg	5.5	1.7	4.16	1.59	-7.17	-10.27	-13.11
Case 4b_0.02	4	5.25	2.07	4.59	1.08	-7.99	-10.29	-10.67
	5	5.48	1.83	3.3	1.29	-6.68	-10.37	-13.06
	6	5.86	1.78	4.23	2.54	-7.35	-12.2	-13.1
	Avg	5.53	1.89	4.04	1.64	-7.34	-10.95	-12.28

Table .5: The table shows the overall percentage change of antenna gain solutions calculated using equation 5.6, and it is the complement of Table .4

.3 Logi_cal Option3

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
Case 1_0.05	3	66	59	59	73	72	84	84
	3	66	67	69	77	73	80	93
	3	64	70	67	73	70	83	89
	4	63	63	60	67	67	81	72
	4	66	64	68	67	65	74	92
	4	67	59	65	71	60	84	76
	5	66	57	55	58	57	73	85
	5	68	70	53	71	70	78	94
	5	66	58	61	64	69	73	90
	6	60	58	61	57	65	69	88
	6	69	55	68	60	79	80	69
	6	66	55	55	66	64	76	90
	Avg	64	59	60	64	70	78	86
case_3a	3	91	82	100	91	52	60	43
	3	93	86	97	91	60	63	59
	3	95	90	97	93	69	77	80
	4	91	81	102	100	38	43	28
	4	92	86	98	103	50	65	39
	4	99	90	98	88	86	81	96
	5	94	83	101	108	46	55	41
	5	98	85	101	84	67	61	51
	5	98	96	87	93	86	84	70
	6	95	86	104	85	34	77	35
	6	96	88	104	87	57	87	48
	6	97	94	97	88	59	88	85
	Avg	95	87	99	93	59	70	56
	3	79	46	54	55	34	41	26

Continued on next page

Table .6 – continued from previous page

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
	3	81	57	50	58	24	35	32
	3	80	56	57	44	19	30	31
	4	75	43	68	66	48	37	30
	4	82	63	66	56	35	32	46
Case 4a_0.01	4	75	56	64	52	31	37	25
	5	90	48	67	61	34	38	37
	5	91	64	56	68	26	42	43
	5	85	46	79	46	39	42	37
	6	93	62	77	59	29	42	31
	6	103	67	71	41	28	48	39
	6	95	61	79	45	33	50	52
	Avg	86	56	66	54	32	40	36
	3	77	45	52	50	27	37	31
	3	77	52	52	44	30	36	37
	3	79	51	58	45	22	33	25
	4	67	50	57	66	45	37	28
	4	80	67	55	58	26	36	38
Case 4a_0.02	4	80	61	65	49	27	36	26
	5	86	47	65	57	35	36	37
	5	91	63	51	67	29	44	34
	5	86	49	61	47	41	36	29
	6	90	63	72	54	31	43	34
	6	96	66	66	44	38	49	39
	6	85	65	74	43	33	48	48
	Avg	83	57	61	52	32	39	34
	3	87	57	71	66	37	32	31

Continued on next page

Table .6 – continued from previous page

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)
	3	93	62	58	68	35	41	29
	3	86	69	68	52	33	33	25
	4	82	64	79	68	58	36	35
	4	95	73	67	62	32	34	43
Case 4b_0.01	4	91	70	67	62	35	36	24
	5	101	55	70	62	43	39	33
	5	95	67	66	72	25	46	39
	5	94	56	72	63	53	38	29
	6	88	68	85	79	30	39	29
	6	94	74	77	61	34	31	28
	6	96	71	74	62	38	34	42
	Avg	92	66	71	65	38	37	32
	3	86	60	69	64	39	31	34
	3	91	64	64	64	34	36	29
	3	88	68	71	56	33	33	23
	4	85	61	83	63	52	37	31
	4	95	73	70	59	33	36	46
Case 4b_0.02	4	86	70	68	61	35	35	25
	5	98	57	67	67	47	37	29
	5	92	72	60	76	31	44	41
	5	95	59	77	66	48	39	29
	6	87	70	87	67	28	37	31
	6	95	74	70	65	37	32	33
	6	91	75	67	65	35	36	39
	Avg	91	67	71	64	38	36	32

Continued on next page

Table .6 – continued from previous page

N_antennas = 124, nfreq=120, ntimes=10, max_bl_cut = 80								
Number of Redundant baseline groups		15 (43.0 m)	25 (52.6 m)	30 (63.6 m)	35 (63.6 m)	40 (66.9 m)	50 (77.3 m)	60 (87.6 m)

Table .6: The table shows a number of antennas that have improved gains out of 124. These results are for option3 of logical where the k-value is randomized within a given range from k=2 to the k-values specified on the left column of the graph. These k-values are repeated three times which indicate the iterations for the parameters. Iterations are implemented to ensure that we do not conclude the performance of this method based on one result since the k-values are random.