Accurate image reconstruction in radio interferometry

Haoyang Ye

Cavendish Astrophysics Group
and Murray Edwards College

February 2019

This dissertation is
submitted for the degree of Doctor of Philosophy
in the University of Cambridge

Contents

Summary

This thesis is concerned with accurate imaging from radio interferometry data and with subsequent analysis so as to determine source positions and fluxes in the radio sky. The thesis makes proposals and implementations of new algorithms, which substantially improve the accuracy of image products and the results of source extraction. These improvements in accuracy promise to assist scientific research into astronomical objects and phenomena in radio astronomy.

The thesis contains six chapters, beginning with an overview of the imaging process in radio interferometry in Chapter 1.

Chapter 2 focuses on improving the accuracy of source extraction, by utilising the Bayesian methodology. The proposed Bayesian method has been implemented in a software package called '`BaSC`' which uses the Markov Chain Monte Carlo (MCMC) technique. By design it works with intermediate radio interferometry image products, such as dirty images, rather than with reconstructed images. `BaSC` achieves greater precision in source location and better resolving power than mainstream source extraction software such as `SExtractor`, which works with reconstructed images. This finding confirms that reconstructed images are not a true representation of the radio sky, whereas dirty images already contain full information about the observations. This piece of work has been accepted by Monthly Notices of the Royal Astronomical Society (Hague et al. 2018). Chapter 2 is based on this paper, but has been rewritten and expanded.

Based on this conclusion, Chapter 3 seeks to optimise the gridding process so as to make accurate dirty images. Since the Fast Fourier transform (FFT) produces dirty images with a much lower computational cost than the Direct Fourier transform (DFT), a new gridding function is needed which minimises the difference between DFT and FFT dirty images. The 'Least–misfit' gridding function is proposed, so as to minimise the image misfit between the DFT and FFT dirty images, and this is implemented and tested. Given an identical support width, it outperforms the main–stream spheroidal function in the image misfit by a factor of at least 100, it also suppresses aliasing in the image plane better. Aliasing is essentially a part of the image misfit, so there is no need to consider it separately. The least–misfit gridding function, with a support width of 7 and an image cropping rate of 0.5, is recommended for

application to both the gridding and degridding processes, and makes it realistic to achieve the limit of single precision arithmetic in the image misfit and visibility misfit.

With the new gridding function in place, Chapter 4 proposes two novel wide–field imaging algorithms: improved W–Stacking and N–Faceting. The improved W–Stacking method uses a three–dimensional gridding, rather than two–dimensional gridding as in the original W–Stacking method. This renders possible the calculation and application of the correcting function on the n (third) dimension. This improvement greatly increases the accuracy of the FFT dirty image on the celestial sphere, relative to the DFT dirty image. The image misfit is as small as 10^{-8} when using the proposed least–misfit gridding function with a support width of 8, and it further reaches the double precision limit when the support width is increased to 14. For comparison, the image misfit levels achieved by the W–Projection algorithm in `CASA` and the original W–Stacking algorithm in `WSClean` are 10^{-3}, several orders of magnitude worse. In addition, since the number of w–planes required by the improved W–Stacking method is reduced compared to the original method, cutting a significant amount of FFT computational cost. As for the original W–Stacking method, if less w–planes than needed are used, the dirty images and reconstructed images produced will underestimate the fluxes of sources that are located far from the phase centre.

The N–Faceting method involves imaging of multiple n–planes, followed by a three–dimensional deconvolution process, where a position–independent beam is used.

Chapter 5 applies the improved W–Stacking method to two real sets of observational data, comprising one GMRT dataset and one VLA dataset. The dirty images on the celestial sphere and the reconstructed images are shown. The improved W–Stacking method successfully removes non–coplanar effects. For the observation with a larger range of w, improved W–Stacking method is recommended to generate a more accurate image with lower computational cost compared to the original W–Stacking method.

Finally, Chapter 6 sets out conclusions drawn from the present work.

Declaration

This dissertation is the result of my own work, which was carried out in the Astrophysics Group of the Department of Physics, University of Cambridge, between October 2014 and February 2019. During this time I have been supervised by Professor Stephen Gull. From October 2015 Dr. Bojan Nikolic acted as my second supervisor. This dissertation includes nothing which is the outcome of work done in collaboration except as declared in the Summary and specified in the text.

Note on the use of the plural pronoun. Throughout this dissertation, "we" is used in the editorial sense to mean "I", "my supervisors and I". In Chapter 2 it refers also to work done in partnership with my collaborator Peter Hague (Hague et al. 2018).

It is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution. I further state that no substantial part of my dissertation has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution.

This thesis contains 47, 848 words including the summary, tables, footnotes and appendices, but excluding the table of contents, photographs, diagrams, figure captions, the bibliography and acknowledgements. It therefore does not exceed the upper word limit of 60, 000 prescribed by the Degree Committee of the Faculty of Physics & Chemistry.

Acknowledgements

I must first – and gladly – express my sincere gratitude to my supervisors, Prof. Stephen Gull and Dr. Bojan Nikolic, for their continuous support. I am grateful to them for their invaluable guidance, immense knowledge and kind encouragement throughout my studies. As I have an engineering background it was at first difficult to adapt myself to the world of astronomers and astrophysicists, early on in my PhD. Thanks to my supervisors, I gradually picked up the knowledge and skills necessary for my research. Without their support, it would not have been possible for me to conduct this research.

My sincere thanks goes also to Dr. Peter Hague. Our collaboration in research has proved to be a highly enjoyable journey. The insightful guidance he has provided, and the discussions we had, helped me very much in developing new research skills and generating new research ideas. His positive attitude towards life has had a great impact on me.

I thank Dr. Sze Meng Tan for enlightening me in research and programming. Much of my work is an extension of his thesis work undertaken more than 30 years ago. I also thank Prof. John Skilling for his useful comments on my research.

I am also very grateful for the support provided by Dr. Dave Green. He has been most helpful in solving LaTeX–related problems. This thesis was written, in fact, using his LaTeXtemplate; I did not change the colour, as I like the default green very much. He also provided me with the calibrated wide–field data used in this thesis.

I thank Dr. Richard Saunders for mentoring me at Astrophysics group. He helped me to face challenges both in research and life with a stronger heart. I very much appreciate his insightful comments and encouragement.

I thank my fellow colleagues for stimulating discussions, happy social times and sincere encouragement. In particular thank Dr. Sonke Hee for his inspirational chats, Dr. Do Young Kim for her positive vibes, Dr. Aglae Kellerer for her wise suggestions, Dr. William Barker for his continuous support, and Dr. James Luis for his encouragement in running. My special thanks go to Greg Willatt for his assistance on operation systems, software installation and many many more IT related problems which he kindly solved for me.

My college has supported me continually since my arrival at Cambridge. This support has

helped me to become a better–rounded individual. I am very grateful that I was admitted to this female–only college. It taught me to believe in my own abilities, set strong but realistic goals, keep challenging myself, and never give up. This experience will assist me in any positions of leadership in my future career. My thanks go especially to my college tutor, Ms. N. C. Cavaleri, who has been a great supporter, listener and advisor. I also thank Dr. Yifu Wang particularly for her consistent support and companionship.

I thank my friends whom I have made both in the UK and back in China. I am very grateful for their love and support. Their knowledge and experience have inspired me to achieve my full potential and be, I hope, a warm–hearted person at the same time.

I owe immeasurable thanks to my parents for supporting me with their unconditional love and care. I believe that I have the best parents in the whole world, and I love them very very much. They have inspired and encouraged me to have the strength and power to be true to myself. Thank you very much, Laoba and Laoma!

Chapter 1

Introduction

This thesis develops methods for improving the accuracy of imaging and source extraction in radio interferometry. Good understanding of the fundamental mechanism and imaging process in radio interferometry is necessary in order to make progress. This work combines distinct areas of science, and some readers are likely to be engineers or computer scientists, who often lack in-depth knowledge of radio astronomy. We therefore begin by outlining the history of radio interferometry in Section 1.1, followed by its basic working mechanism in 1.2. The imaging procedures, proceeding from raw observed data, are presented in Section 1.3. Image reconstruction methods are briefly introduced in Section 1.4. An outline of the thesis is set out in 1.5.

Readers interested in only one of the particular chapters in this thesis are recommended to read the present chapter, then proceed directly to the chapter of interest.

1.1 A brief history of radio interferometry

Radio astronomy is a subfield of astronomy which emerged after the first detection of radiation from the Milky Way by Jansky (1932, 1933, 1935). Following this discovery, more observations were conducted using a single antenna. Later, Reber (1940) built the world's first single parabolic radio telescope, and his observations confirmed Jansky's discovery. The resolution of the telescope (or angular resolving power) is roughly proportional to the observing wavelength divided by the diameter of the parabolic dish, so that a more detailed observation requires a larger dish. For example, the resolution achieved by Reber (1940) using his 9–metre dish at

160 MHz is approximately 12°. A resolution of 1.2° therefore requires a 90–metre dish, which was impractical at the time.

Radio interferometry was invented to achieve high resolution without the need to build an impossibly large dish. Instead of a single antenna, a radio interferometer consists of multiple antennas at separate locations. The distance between any two antennas is called a 'baseline'. The connected arrays of antennas can be considered as a single radio telescope with a large 'synthesised' aperture. The diameter of the synthesised aperture is the maximum baseline. This aperture records data only at the antennas. The desired high (small angle) resolution can be achieved when the baseline is sufficiently large. Sullivan (1991) stated that the radio interferometer constituted a major advance in radio astronomy.

Ryle et al. (1948) built a simple interferometer with two antennas in order to observe the radiation from the sun. From then on, radio interferometry began to develop. Voltage–multiplier circuits were introduced to implement the multiplication of two signals received by two antennas, and this was followed by time–averaging circuits. The resulting combined circuitry is able to implement correlation calculations, and is usually termed a 'correlator'. The correlator output is called the 'spatial coherence', 'visibility function' or 'visibilities'. Visibilities are complex and are dependent on the baselines, observation frequency and the integration time. They can be used to reconstruct the sky brightness distribution.

Earth–rotation aperture synthesis was introduced to take advantage of the rotation of the Earth so as to increase differing baseline spacings during observations (Ryle & Neville 1962). The One–Mile Telescope comprising three dishes was built at Cambridge as the first telescope to use this technique (Elsmore et al. 1966). Earth–rotation synthesis has since been applied in modern radio interferometry. A classic example of a modern radio interferometer is the Very Large Array (VLA) in New Mexico (Thompson et al. 1980; Napier et al. 1983), which is a US National Radio Astronomy Observatory instrument. This large array comprises 27 steerable dishes each of diameter 25m, distributed on Y–shaped tracks over 35km. Some of the simulated visibility data in this thesis is based on the VLA configuration. We use the simulated data to test the performance of software and algorithms.

As with the VLA, many radio interferometers have been built for better sensitivity, resolution, faster data processing and differing ranges of the observing frequency (Thompson et al. 2017, Chap. 1.3). With the development of very–long–baseline interferometry (VLBI) techniques (Walker 1999), interferometers with enormous baselines have been constructed to achieve high resolution. One example is the Very Long Baseline Array (VLBA) having 10 antennas located across the United States. These antennas are operated remotely by the Array Operations Center (AOC) in Socorro, New Mexico (Kellermann & Thompson 1985; Napier et al. 1994). This instrument has a maximum baseline of 8000 km, achieving a resolution as

high as 1 milliarcsecond (mas) at centimetre wavelengths. In terms of the frequency range, the Atacama Large Millimeter Array (ALMA) was constucted to operate at high frequencies from 30 to 950 GHz (Brown et al. 2004), whereas the Low–Frequency Array (LOFAR) was designed to operate at low frequencies from 10 to 240 MHz (van Haarlem et al. 2013). An interferometer project currently under construction is the Square Kilometre Array (SKA), which will become the most sensitive radio telescope project (Hall et al. 2008) with a sensitivity 100 times greater than the VLA. It will also have a very wide frequency range, from as low as 100 MHz to 25 GHz (Schilizzi 2004).

Data collected by these modern interferometers is usually processed by software to implement the data reduction process, and the imaging process is then applied to produce the desired images. The software package known as Common Astronomy Software Applications (`CASA`[a]) is a mainstream package used to process both radio single–dish and interferometry data (McMullin et al. 2007). Many simulated data in this thesis have been generated using `CASA` .

Radio interferometry still has a considerable advantage over single–antenna telescopes today. Whereas SKA can achieve a minimum resolution of 1 mas at 20 GHz (Schilizzi 2004), the world's largest single–dish radio telescope, FAST (Five–Hundred–Meter Aperture Spherical Radio Telescope (Nan et al. 2011)), with a diameter of 500 meters, can achieve only a resolution of 25 arcsec at its highest operation frequency 5 GHz.

1.2 Fundamentals of synthesis imaging

The fundamental technique used in modern radio interferometry to recover the sky brightness distribution from visibilities is known as 'synthesis imaging' or 'synthesis mapping'. The radio sky imaged by an interferometer can be termed a map or an image. In this thesis, we use these terms interchangeably.

The relation between the visibility data V and the sky brightness distribution I is derived clearly by Clark (1999) and Thompson (1999). Following the (u, v, w) and (l, m, n) coordinate systems defined in Thompson (1999), this relation can be expressed as:

$$V(u,v,w) = \int\int \frac{\mathrm{d}l\mathrm{d}m}{\sqrt{1-l^2-m^2}} I(l,m) \exp[-i2\pi(ul + vm + w(\sqrt{1-l^2-m^2} - 1))] \quad (1.1)$$

Each angular coordinate (l, m, n) describes a location on the celestial sphere, where $n = \sqrt{1 - l^2 - m^2}$ and l and m are direction cosines between -1 and 1. Since the synthesised image is two dimensional, each value of (l, m) represents a location on the tangent plane projected from the celestial sphere. The tangent plane is perpendicular to the n axis. Here, (u, v, w) are baseline coordinates on the Earth in units of wavelength, with the w axis pointing outward from

[a] http://casa.nrao.edu/

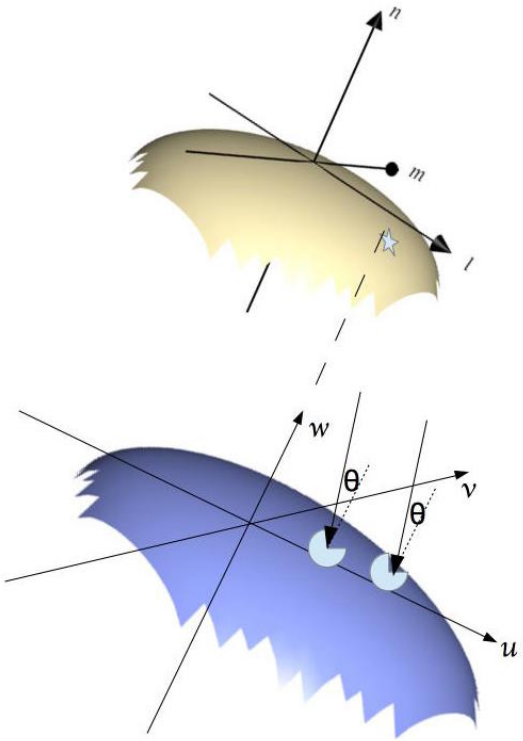


Figure 1.1: Coordinates of celestial sphere and baselines. The direction cosines (l, m) form the two–dimensional tangent plane, which is perpendicular to the n axis. The baseline coordinates are on the Earth with the w axis pointing directly outward from the phase centre. Two antennas are on the same uv–plane receiving radiation emitted from a bright source at an angle θ.

the phase centre, or the centre of the field of view. These sets of coordinates are shown in Figure 1.1.

The source element $\mathrm{d}\Omega$ of the surface of the celestial sphere S is given by

$$\mathrm{d}\Omega \equiv \frac{\mathrm{d}l\mathrm{d}m}{n} = \frac{\mathrm{d}l\mathrm{d}m}{\sqrt{1-l^2-m^2}}$$

Consequently, the flux at a given location (l, m) is equivalent to $\frac{I(l,m,n)}{n}$.

If the observation field is very small and is close to the phase centre, so that l and m are very small and $n = \sqrt{1-l^2-m^2} \approx 1$, then $w(n-1) \approx 0$ can be neglected. The visibility function can then be written as a function of u and v alone:

$$V(u,v) \propto \int\int \mathrm{d}l\mathrm{d}m I(l,m) \exp[-i2\pi(ul+vm)] \tag{1.2}$$

Thus, the sky brightness can be obtained by performing an inverse two–dimensional Fourier transform on the visibility function $V(u, v)$:

$$I(l, m) = \int \int \mathrm{d}u\mathrm{d}v V(u, v) \exp[i2\pi(ul + vm)] \tag{1.3}$$

Observation fields with considerable values of ls and ms are called 'wide fields', where the term $w(n-1)$ can no longer be neglected, in which case the two–dimensional Fourier transform relation is not valid. Imaging of wide–field observations will be discussed in Chapter 4.

We have not yet considered the 'primary beam' $A(l, m)$, which is defined as the response beam pattern of the antenna, describing its sensitivity at different directions. If we assume that all antennas share the same primary beam, the observed sky image $I(l, m)$ recovered from Equation 1.1 will become the product of the real sky $I^{\mathrm{true}}(l, m)$ and this primary beam $A(l, m)$:

$$I(l, m) = I^{\mathrm{true}}(l, m)A(l, m) \tag{1.4}$$

The primary beam is typically a two–dimensional Gaussian function rather than a constant, giving rise to flux inaccuracy, especially at the edges of the image.

1.3 Synthesis imaging

Before being used for imaging, the raw visibility measurements observed need to be edited and calibrated (Fomalont & Perley 1999). Editing is performed to identify and remove data that is severely corrupted. For example, visibilities obtained by baselines involving a malfunctioning antenna will be 'flagged' and not be used in subsequent data processing stages. Calibration is intended to correct atmospheric and instrumental errors which affect observations. The edited and calibrated visibilities are used for imaging according to the two–dimensional Fourier transform relation explained in Section 1.2.

1.3.1 Visibility sampling and weighting

To apply the Fourier transform relation in Equation 1.3 it is necessary to measure the visibility function for all baseline spacings. But a radio interferometer usually has a very limited number of antennas fixed at various locations on the Earth, making full sampling of baselines impossible. The rotation of the Earth was exploited to increase baseline samples, but we still obtain an incompletely sampled synthesised aperture, which is defined as '*uv*–coverage'. The *uv*–coverage is affected by the declination of the observations and their duration.

The *uv*–coverage can be plotted by putting all baselines of the given interferometer during the observation on the *uv* plane. Figure 1.2 shows two examples of the *uv*–coverage plots produced by `CASA`. The top *uv*–coverage plot is based on a VLA A–array 10-second simulated

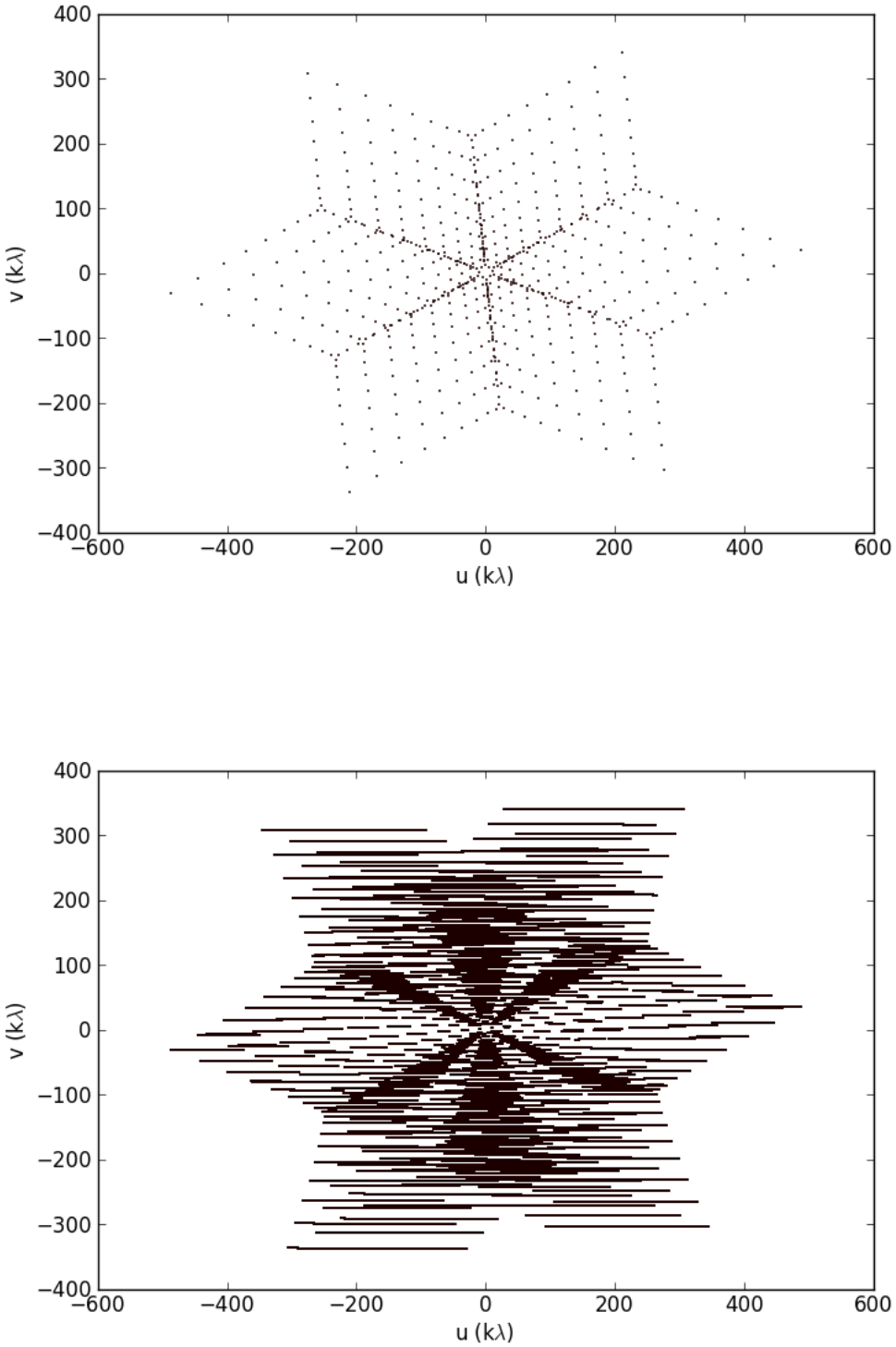


Figure 1.2: *uv*–coverage for a VLA A–array 10-second observation (top) and a VLA A–array 5-hour observation (bottom) with zero declination. The simulated observation wavelength λ is 7.5 centimetre, and the u and v axes are in units of 1000λ, written as $k\lambda$.

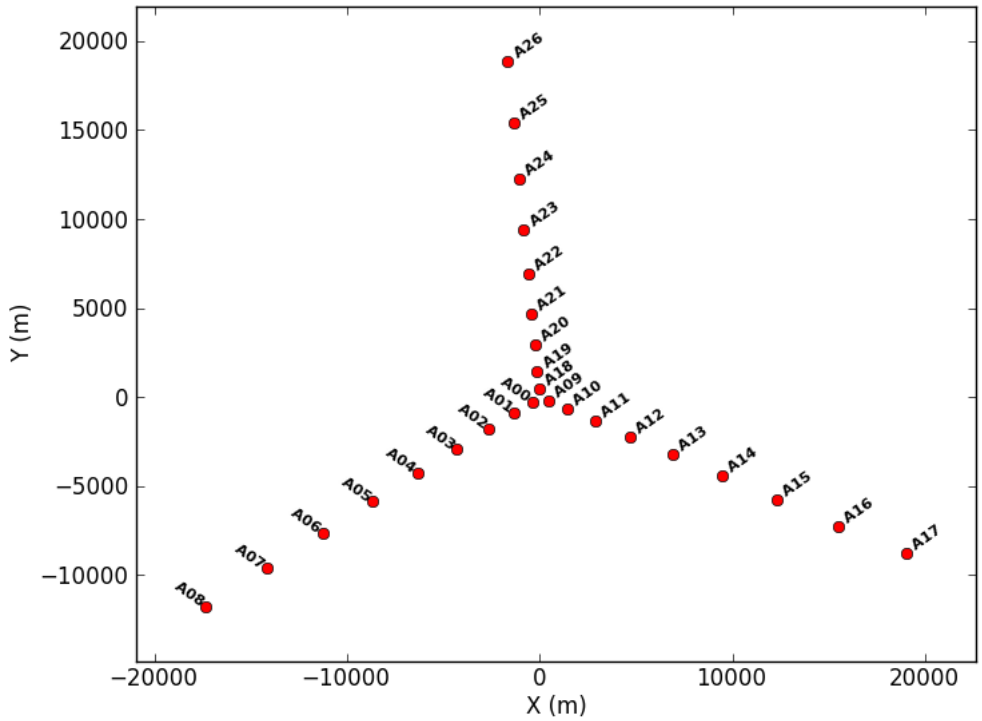


Figure 1.3: VLA A–array configuration. The red spots represent antennas and the characters written adjacent to them are the antenna names. The x– and y–axes are centred at the intersection of the three arms of the Y–shaped array.

observation, and the lower plot is based on a 5-hour VLA A–array simulation with zero declination. The simulated single–channel observation wavelength λ is 7.5 centimetre, and the u and v axes are in units of 1000λ, written as $k\lambda$. It can be seen that a longer observation has more uv spacing samples, due to the Earth's rotation. As the Earth rotates, each baseline has a changing projection on the uv plane.

The configuration of the VLA A–array, comprising 27 antennas, is plotted in Figure 1.3, with red spots representing antennas and antenna names written adjacent to them. If the source direction is not at zero declination, each baseline will trace out two arcs in the uv plane; see Figure 1.4 for the uv–coverage from a VLA A–array 5-hour simulated observation with the declination equal to 45°.

Since the observed visibilities V^S is sampled from the full visibility function V at all baselines available during the observation, a sampling function $S(u, v)$ is introduced as

$$S(u, v) = \sum_{k=0}^{M} \delta(u - u_k)\delta(v - v_k), \tag{1.5}$$

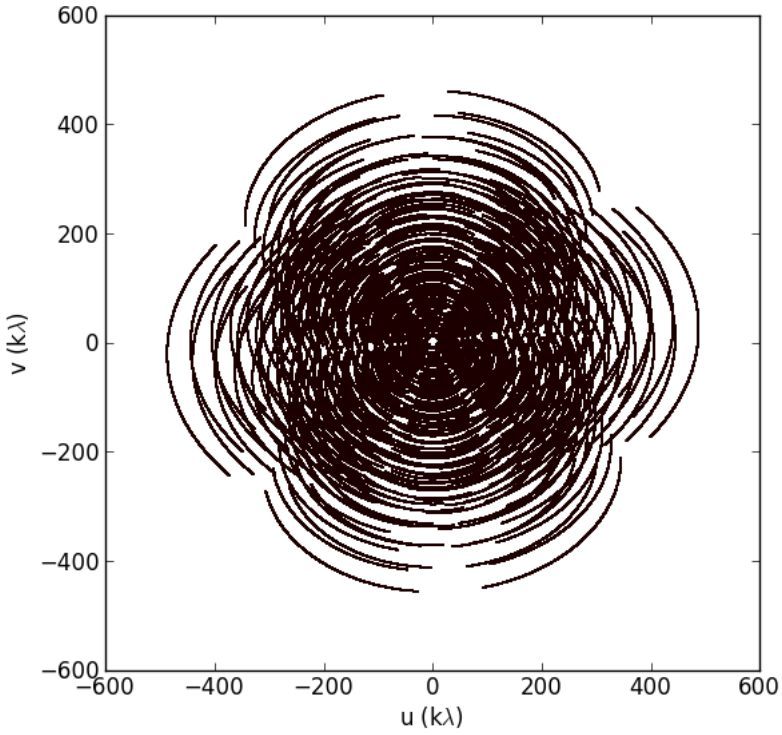


Figure 1.4: *uv*–coverage for a VLA A–array 5-hour observation with a declination of 45°.

where M is the number of baselines. Each Dirac delta function stands for a different baseline sampled on the uv plane. The sampled visibility function V^S is the product of V and S:

$$V^S(u,v) = S(u,v)V(u,v) \tag{1.6}$$

Each visibility data point $V^S(u_k, v_k)$ is given a weight w_k:

$$W(u,v) = \sum_{k=0}^{M} w_k \delta(u-u_k)\delta(v-v_k) \tag{1.7}$$

where $W(u,v)$ is a 'weighting function', designed to improve the dynamic range and fidelity of the reconstructed sky image. Many weighting functions are already embedded in `CASA`, including natural weighting, uniform weighting and briggs weighting.

In this thesis, natural weighting is used for the best signal–to–noise ratio (Thompson 1999) unless otherwise stated. Natural weighting gives the same weight at $w_k = \frac{1}{\sigma_k^2}$ to the visibility data $V^S(u_k, v_k)$, where σ_k^2 is the noise variance of the visibility. Hence the sampled and natural weighted visibility function V^{SW} is written as:

$$V^{SW}(u,v) = \sum_{k=0}^{M} \frac{1}{\sigma_k^2} V(u,v)\delta(u-u_k)\delta(v-v_k) \equiv S(u,v)W(u,v)V(u,v) \tag{1.8}$$

1.3.2 Imaging and gridding

As explained in §1.3.1, the uv–plane is not fully sampled. The inverse direct Fourier transform (DFT) of the sampled and weighted preprocessed visibilities becomes

$$I_D(l,m) = \int\int \mathrm{d}u\mathrm{d}v S(u,v)W(u,v)V(u,v)\exp[i2\pi(ul+vm)] \tag{1.9}$$

We call I_D a 'dirty map' or 'dirty image'. It is not an actual sky image, and it is often referred to as a 'map' in the literature. It can be written as a convolution of the inverse DFT of $S(u,v)W(u,v)$ and $V(u,v)$, according to the convolution theorem:

$$I_D(l,m) = B(l,m) * I(l,m) \tag{1.10}$$

where B is defined as the 'synthesised beam', or 'dirty beam'. The term 'dirty beam' will be adopted in this thesis. It is the inverse DFT of the weighted sampling function.

Since the dirty image and dirty beam are both real image products, they can be written as

$$I_D(\mathbf{x}) = \Re\left[\sum_{k=1}^{M}\frac{V_k}{\sigma_k^2}\exp\left(i2\pi\mathbf{u}_k\mathbf{x}\right)\right] \tag{1.11}$$

$$B(\mathbf{x}) = \Re\left[\sum_{k=1}^{M}\frac{1}{\sigma_k^2}\exp\left(i2\pi\mathbf{u}_k\mathbf{x}\right)\right] \tag{1.12}$$

where $\mathbf{u} = (u,v)$ and $\mathbf{x} = (l,m)$. We denote $V^S(u_k,v_k)$ as V_k.

The inverse Fast Fourier Transform (FFT, Cooley & Tukey (1965)) is usually adopted, rather than an inverse DFT, in order to save computational time. To use FFT, the irregularly sampled visibility data must first be interpolated onto uniform rectangular grids. This procedure is called 'gridding'.

The most straightforward way of gridding is to assign each item of visibility data to its nearest grid point, but this procedure results in significant artefacts and a considerable difference between the DFT and FFT dirty images. Figure 1.5 shows a DFT dirty image, a FFT dirty image using nearest–point interpolation method, and their image misfit based on simulated data. No primary beam correction was applied. The image size is substantially smaller than the primary beam, so we can neglect the effects of attenuated source fluxes here. The data used derive from a VLA A–array 10-second single–channel observation simulated on `CASA` at 4 GHz. The five point sources are 2 Jy[b] with their locations at $(0'',0'')$, $(1.8'',1.8'')$, $(1.8'',-1.8'')$, $(-1.8'',1.8'')$, $(-1.8'',-1,8'')$ respectively. The top left image is the DFT dirty image of the simulated data, and the top right image is the FFT dirty image with every visibility placed at its closest grid point. The difference between these two images is displayed at bottom

[b] Jy, or Jansky, is a spectral flux density unit. 1 Jy is equivalent to $10^{26}\mathrm{Wm}^{-2}\mathrm{Hz}^{-1}$.

left, showing obvious errors and increasingly significant noise towards the edges of the image. In this thesis we refer to the difference between the DFT dirty image and the FFT dirty image as the 'image misfit'. The Root Mean Square (RMS) value of the image misfit is as large as 5.67×10^{-2}, and over the central one–third of the plot it is 1.85×10^{-2}.

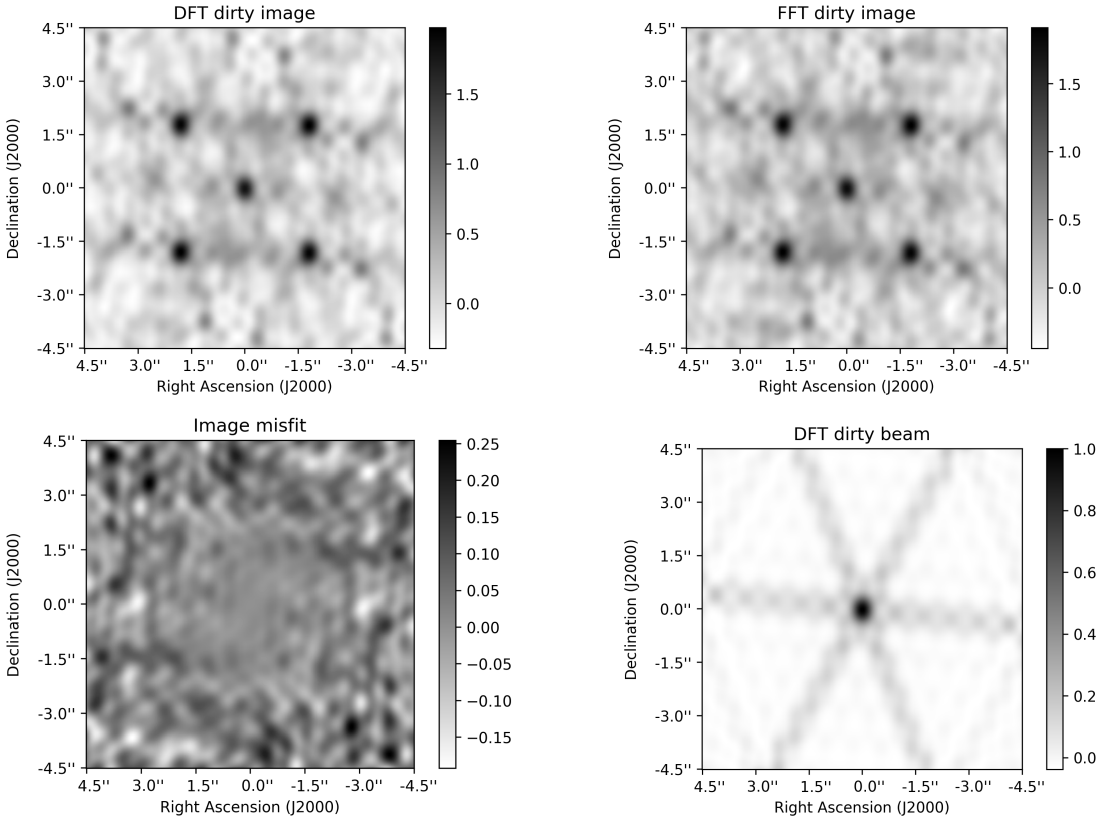


Figure 1.5: The top left image is the DFT dirty image of the simulated data, and the top right image is the FFT dirty image with every visibility placed at its nearest grid point. The difference between these two images is shown at bottom left with obvious artefacts and increasingly significant noise towards the image edges. The dirty beam is shown at the bottom right.

The most common method used is to convolve the visibility data with a gridding function. The gridding function $C(u, v, w)$ should satisfy two conditions, as follows: (Greisen 1979).

1) The gridding function is separable: $C(u, v, w) = C_u(u)C_v(v)C_w(w)$, where C_u, C_v and C_w are usually chosen as the same function.
2) The gridding function is real and symmetric about the origin: $C(k-u) = C(u-k)$. The one–dimensional form $C(u)$ is used for simplicity from now on.

A spheroidal function is the most widely–used gridding function in use at the present time. Since a convolution with an infinite long function will result in infinite convolution calculation time, the gridding function C should be confined by a pillbox with a chosen width.

Figure 1.6 demonstrates a typical gridding process using a gridding function $C(u)$ with a support width $W = 8$. Here W is not required to be an even number; the odd number case will be discussed in Chapter 3. An off–grid visibility value $V(u_0)$ at position u_0 is marked in the figure by the blue cross. This will be weighted and assigned to 4 grids on either side, with weights taken from $C(u)$. The weight for each grid can be written as $\{C([u_0] - u_0 + r)\}$ where $r = -3, -2, ..., 4$, and $[u_0]$ is the largest integer not exceeding u_0. These weights are marked by red dots in the figure. As this process is repeated for all visibility data, the assigned visibility values on each grid will be summed. After that, the visibility data is gridded into uniform rectangular grids.

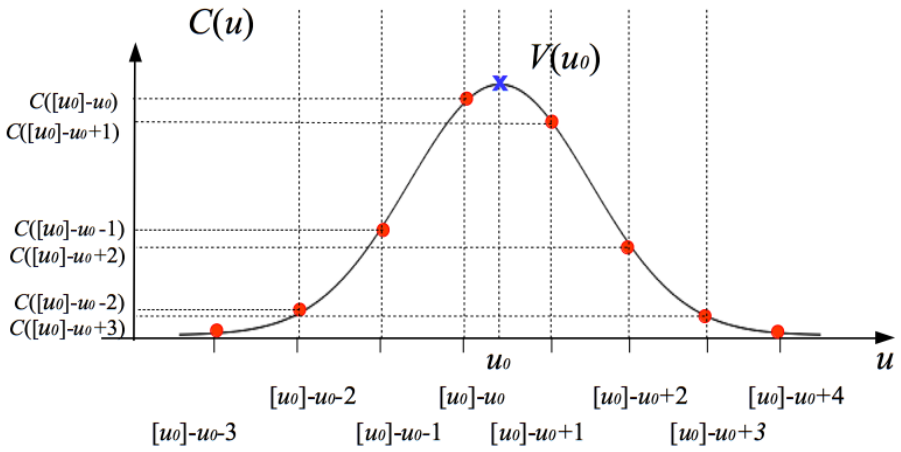


Figure 1.6: A demonstration of the usage of an one–dimensional gridding function $C(u)$. For given visibility $V(u_0)$ at position u_0, the grid points on both sides of u_0 within width 4 will have weights assigned. The weights for these points are $\{C([u_0] - u_0 + r)\}$ where $r = -3, -2, ..., 4$, and $[u_0]$ is the largest integer not exceeding u_0. The weight for each grid is marked by a red dot.

An inverse FFT is applied to the gridded visibility data. After image cropping and multiplication with the 'correcting function' on the image plane, the FFT dirty image is obtained. The operation of image cropping is to remove the noise at the image edges, which will be discussed in detail in Chapter 3.

In summary, the operations giving the dirty image using the gridding and inverse FFT are:

1) Grid the visibility with a chosen gridding function on the uv plane.
2) Apply the inverse FFT to the gridded visibility data to obtain an image.
3) Crop the image and multiply the correcting function by the cropped image.

The choice of gridding function and the calculation of the correcting function are considered in Chapter 3.

1.4 Deconvolution

Based on Equation 1.10, the true sky brightness $I(l, m)$ can be recovered by deconvolution from the dirty image $I_D(l, m)$. As is explained in (Cornwell et al. 1999; Venkata 2010), for observations with an incomplete sampled uv–plane a non–linear deconvolution method is required to reconstruct the true sky brightness. After that a 'primary beam correction' (mentioned earlier) is usually necessary, following Equation 1.4, to correct the attenuated source fluxes in the reconstructed image.

In this section, two mainstream deconvolution methods are introduced. The first is the CLEAN algorithm. CLEAN was proposed by Högbom (1974), and was followed by many variants, such as Clark CLEAN (Clark 1980), Cotton–Schwab CLEAN (Schwab & Cotton 1983) and Multi–scale CLEAN (Cornwell 2008). The second deconvolution method considered here is the Maximum Entropy Method (MEM) (Gull & Daniell 1978; Gull & Skilling 1984b; Narayan & Nityananda 1986; Cornwell & Evans 1985; Gull 1989; Skilling 1989). These two methods will be implemented in Chapter 4 as tests of wide–field imaging methods.

1.4.1 The CLEAN algorithm

CLEAN proceeds on an assumption that there exist only point sources in the field of view. It uses an iterative approach to find a list of flux densities and locations of these point sources, or 'CLEAN components'. Apart from Högbom's CLEAN (Högbom 1974), variants usually consist of two cycles: the major cycle on the visibility plane, and the minor cycle on the image plane.

Högbom CLEAN (Högbom 1974) has only minor cycles. In each iteration, a scaled dirty beam is subtracted from the dirty image (or the subtracted dirty image) at its brightest position. The subtracted dirty image is defined as the 'residual'. The position and flux of this brightest pixel, which is usually referred to as a 'CLEAN component', is recorded and added to an image model which begins as an all–zero two–dimensional matrix. The iteration is halted when the brightest pixel on the residual image is below a chosen threshold, or the number of iterations exceeds a chosen maximum number. The final residual image should contain mainly noise. Thus, the reconstructed image, or 'CLEANed' image, can be obtained by convolving the most updated image model with a 'CLEAN restoring beam' (usually a Gaussian) plus the final residual image.

To save computational time, Clark CLEAN (Clark 1980) uses a patch of the dirty beam containing the highest sidelobe, rather than the full beam, during the minor cycle. The minor cycle is similar to the Högbom CLEAN, which is essentially a process that finds a list of bright pixels on the dirty image or residuals and then adds them to the image model. Unfortunately the residuals produced by the minor cycles are not accurate, because the incomplete dirty beam is used. The residuals should therefore be recalculated by subtracting from the dirty image the convolution of the image model obtained with the full dirty beam. According to the convolution theorem, this convolution becomes a multiplication of the visibility model and the weighted sampling function, which are the Fourier transforms of the image model and full dirty beam respectively. In this way an accurate residual can be obtained by subtracting the inverse Fourier transform of the multiplication result from the dirty image. The resulting residual image will be used to start a new minor cycle.

The Fourier transform in the major cycle of Clark CLEAN is usually applied through a FFT in order to save time, which will result in uniformly sampled visibilities on rectangular grids. Cotton–Schwab CLEAN (Schwab & Cotton 1983) uses instead a degridding/FFT route to obtain the visibility model at the original sampled values of (u, v, w). Next, a 'visibility plane residual' is obtained by subtracting the visibility model from the original visibility data directly, which will then be transformed to the image.

The process of degridding/FFT aims to recover visibilities at the original irregularly sampled baselines from an image model. This process is usually performed via the correlation between the FFT of the image model and the gridding function introduced earlier. Since gridding and degridding are transpose operations, the degridding operations can be summarised as follows:

1) Multiply the model image by the correcting function.
2) Pad zeros to the outer edge of the corrected model image, which is the opposite operation of image cropping during gridding.
3) Apply FFT to the corrected image.
4) Correlate the gridding function with the FFT results to obtain the visibility model.

The degridding/FFT route is also implemented in the self–calibration process (Cornwell & Fomalont 1999), where the observed visibilities are corrected based on degridded visibilities from image models obtained from the deconvolution process.

Similarly, gridding and the inverse FFT substitute the inverse DFT inside the Cotton–Schwab CLEAN method.

1.4.2 Maximum entropy method

For a given interferometer observation, there a large set of 'feasible' sky images is consistent with this observation, because of the incomplete uv sampling. In MEM, an 'entropy' function

is introduced that is to be maximised so as to find the most probable nonnegative image under some predefined constraints from the feasible images (Gull & Skilling 1984a).

The term 'entropy' here is not immediately related to that defined in thermodynamics or information theory. The entropy of a given image $I(l, m)$ is defined as (Narayan & Nityananda 1986):

$$S = \int \int f([I(l, m)])\mathrm{d}l\mathrm{d}m \tag{1.13}$$

where f is defined as the entropy function. The entropy function has different forms, which all guarantee nonnegative intensity value on each pixel. In MEM application to radio image reconstruction in Gull & Daniell (1978), the entropy function is chosen as

$$f = -I \log I \tag{1.14}$$

If prior knowledge is considered, the entropy function can be generalised to (Gull & Skilling 1984a; Gull & Skilling 1984b) to

$$f = -I \log(I/I_0) \tag{1.15}$$

where I_0 is the initial model. If no prior information is provided, I_0 would be a flat image and Equation 1.14 is the result.

In MEM, a constraint is defined via a χ^2 statistic to optimise the image model such that it is consistent with the observed data as closely as possible. Supposing that the noise in V_k is Gaussian with a variance σ_k^2, χ^2 is written as:

$$\chi^2 = \sum_k^M \frac{|V_k - V_k^{'}|}{\sigma_k^2} \tag{1.16}$$

where V' comprises the degridding results from the current image model and M is the number of visibilities.

In 'Historic MaxEnt', proposed by Gull & Daniell (1978), an iterative procedure is used to modify the image models. Iterations continue until the image entropy is maximised subject to $\chi^2 = M$. The prior image, or the starting image model, is usually chosen as a flat image or as the dirty image. A Bayesian version of MEM, or 'Classic MaxEnt', was given in Gull (1989). historic and Classic MaxEnt are both implemented in the licensed software package `MemSys5` (Gull & Skilling 1991).

Cornwell & Evans (1985) also proposed a MEM algorithm using an objective function containing both an entropy function and χ^2 with Lagrange multipliers (Cornwell & Evans 1985). The algorithm uses a similar iteration process, which stops once χ^2 reaches an acceptable limit. This algorithm is available in Astronomical Image Processing Software (`AIPS`) (Wells 1985).

Sky images reconstructed using these two methods are still not true representations of the radio sky. Information deriving from reconstructed images, such as source flux and position,

must therefore be considered with extra care. A Bayesian method to abstract such information from dirty images will be detailed in Chapter 2.

1.5 Thesis outline

The aim of this thesis is to develop accurate imaging methods and implement accurate source extraction in radio interferometry. The hope is that the resulting improvements in accuracy will further scientific research on astronomical objects and phenomena in radio astronomy.

An outline of this dissertation now follows.

Specifically, the aim of the research project detailed in Chapter 2 was to overcome the limitation that main–stream source extraction methods used in radio interferometry work only on reconstructed images[c]. Based on the idea that reconstructed images are not a true representation of the sky, whereas the dirty image already contains all the information form the observations, we have proposed a method which extracts source information using intermediate image products, such as dirty images, rather than reconstructed images.

Chapter 2 first provides an overview of existing astronomical source extraction methods. After introducing Bayes' theorem, a Bayesian multiple–source model is constructed and implemented via Monte Carlo Markov Chain (MCMC) using software called `BaSC`, so as to find the position and flux information of sources in the image plane. This software is intended to work with intermediate image products in radio interferometry, namely dirty images, dirty beams and primary beams. We display comparisons between the results of using our software and other mainstream source extraction software on simulated data. We also discuss the resolving power for the position in this proposed method. That item of work has been accepted by Monthly Notices of the Royal Astronomical Society (Hague et al. 2018). This chapter is based on that paper, but has been rewritten and expanded.

The gridding and degridding process during imaging is very important for the image quality but is also computationally very expensive. The gridding function should therefore be chosen with care, but little investigation has been conducted on that since the 1980s. As the gridding function is adopted in using FFT in place of DFT to reduce the computation time, we aim to find a gridding function which can minimise the loss of image accuracy. We sought gridding a function which can minimise the difference between the DFT and FFT dirty images. This gridding function is also used during the degridding process to produce accurate visibility models, especially in the image reconstruction and self–calibration steps.

In Chapter 3, after a review of the gridding process with different gridding functions, we propose a gridding function called the 'Least–misfit gridding function', which aims to minimise

[c]More details can be found in Chapter 2

image errors due to the replacement of DFT by FFT, thereby maximising the accuracy of the FFT dirty image. The results of experiments using simulated data are shown which demonstrate this improvement in accuracy, as well as giving superior aliasing–suppression performance than current mainstream spheroidal functions, which were proposed to reduce aliasing effects. This chapter will be abbreviated and submitted to Monthly Notices of the Royal Astronomical Society.

In the more complex wide–imaging problem, with non–coplanar arrays, the basic two–dimensional Fourier inversion relation between the visibility data and the sky brightness no longer holds. Chapter 4 sets out the results of research into new methods for improving the accuracy of wide–field imaging. An overview of wide–field imaging problems is first provided. After discussion of some existing wide–field imaging methods, two further methods are proposed. The first is based on an existing method called 'W–Stacking', and the second is named 'N–Faceting'. Compared to the existing method, the improved W–Stacking method considerably improves the accuracy of the celestial sphere FFT dirty image, judged in terms of the difference from the DFT dirty image. The N–Faceting method involves the imaging of multiple n–planes followed by a three–dimensional deconvolution process, where a position–independent beam is used rather than a single tangent plane beam. The performance of these two methods is demonstrated and compared against some common current methods.

We also test the improved W–Stacking method on calibrated real observations, and find the method to be highly promising for real applications. In Chapter 5, dirty images on the celestial sphere and reconstructed images are both displayed.

Chapter 6 sets out the conclusions of the original part of this thesis.

CHAPTER 2

BAYESIAN SOURCE DISCRIMINATION IN RADIO INTERFEROMETRY USING BASC

Source extraction is a process that generates catalogs of astronomical objects, and abstracts features such as locations, shapes and brightnesses from observations. Catalogs can be used to conduct statistical analysis, so as to test hypotheses and models (Maddox et al. 1990).

Prior to electronic computers, source extraction was performed by eye. It was extremely time–consuming and lacked accuracy. Today, computing software and systems automate the source extraction process. `SExtractor` (Bertin & Arnouts 1996) is a widely used software package for source extraction. It was designed to work with optical observation data, typically images from charge–coupled devices (CCD). `SExtractor` is based on the idea that the value at each image pixel is the sum of background signal and object brightness within the pixel. Although not designed for radio interferometry, `SExtractor` has been used to process CLEANed images since the first attempt by Bondi et al. (2003). Unfortunately it is unable to work on intermediate image products, such as dirty images. Existing source extraction packages designed for radio interferometers, such as SAD (Search and Destroy) in the software package AIPS, SFIND (Hopkins et al. 2002) in the software package MIRIAD (Sault et al. 1995), PyBDSF (Python Blob Detection and Source Finder) (Mohan & Rafferty 2015) and the most recently published source extraction package `Aegean 2.0` proposed by Hancock et al. (2018), also work only on reconstructed images and not on dirty images.

Although `SExtractor` and these other packages are able to work on reconstructed images, their accuracy is in doubt. This is primarily because those images are not true representation

of the sky brightness. The accuracy of results deriving from them is therefore difficult to determine. This problem is more severe when a complex synthesised beam is present. The reconstruction process also requires intensive computational and human efforts.

CLEAN produces sky models by proposing CLEAN components, and inserts a restoring beam at the position of each component. When working on a CLEANed image, SExtractor's resolving power is limited by the CLEAN restoring beam. Therefore, SExtractor is unsuitable if the structures under investigation have a smaller scale. An example occurs in Trakhtenbrot et al. (2017), in which further investigation is necessary of the differential rotation of galaxies discovered from CLEANed velocity maps at a scale comparable to the size of the restoring beam. Source extraction packages working on CLEANed images are unsuitable. Experiments conducted in §2.5.2 will confirm that the resolving power of SExtractor is limited by the size of restoring beam.

A source extraction package designed specifically for radio interferometry is needed that does not limit itself to working with reconstructed images. Its angular resolution should be smaller than the size of the CLEAN restoring beam.

This chapter implements and tests a Bayesian source extraction method that utilises the Markov Chain Monte Carlo (MCMC) process. This method takes as input dirty maps, dirty beams and primary beams. The method has been integrated into a Python public library called BaSC [a]. BaSC is tested on simulated data, and its capability for source discrimination is compared to SExtractor and Aegean 2.0. The result is that BaSC better distinguishes nearby sources than SExtractor or Aegean[b]. Much of the work in this chapter has been submitted to the journal Monthly Notices of the Royal Astronomical Society as Hague et al. (2018).

This chapter is arranged as follows. In Section 2.1 we introduce Bayes' theorem and the Bayesian multiple point–source model as the foundation for the new method. Section 2.2 sets out how to implement the MCMC process so as to detect the source fluxes and positions. In Section 2.3, a clustering process is introduced to find source locations and fluxes from MCMC results. Three clustering algorithms will be discussed in detail: Density–based Spatial Clustering of Applications with Noise (DBSCAN), Ordering Points To Identify The Clustering Structure (OPTICS) and Hierarchical DBSCAN (HDBSCAN). Section 2.4 derives the source detection accuracy, or Bayesian resolving power, of the method. Section 2.5 compare the source discrimination performance of BaSC, SExtractor and Aegean based on the same simulated data. Further experiment confirms the formula for the Bayesian resolving power. Conclusions are presented in Section 2.6.

[a] https://github.com/petehague/BASC

[b] In this chapter, Aegean is short for Aegean 2.0

2.1 Bayes' theorem and the Bayesian point–source extraction model

A brief introduction to Bayes' Theorem is now given. For a given dataset $\boldsymbol{I}$, under hypothesis H we can build a parameter–based model with a set of parameters $\boldsymbol{\theta} = \{\theta_0, \theta_1, ...\}$. According to Sivia & Skilling (2006), Bayes' theorem can be written as

$$P(\boldsymbol{\theta}|\boldsymbol{I}, H) = \frac{P(\boldsymbol{I}|\boldsymbol{\theta}, H)P(\boldsymbol{\theta}|H)}{P(\boldsymbol{I}|H)} \tag{2.1}$$

where $P(\boldsymbol{\theta}|H)$ denotes the prior probability for the parameters, given the hypothesis H but in the absence of the data, and $P(\boldsymbol{\theta}|\boldsymbol{I}, H)$ is the posterior probability after the dataset has been incorporated. The term $P(\boldsymbol{I}|\boldsymbol{\theta}, H)$ is known as the likelihood, and $P(\boldsymbol{I}|H)$ is the marginal likelihood, or 'evidence'.

The principles of Bayesian inference are also set out clearly in Hobson & McLachlan (2003), but their method of Bayesian source extraction is based on an image–plane model, and therefore works only on reconstructed images. In contrast to existing Bayesian source extraction methods (Hobson & McLachlan 2003; Lochner et al. 2015; Savage & Oliver 2007), we introduce a method that takes into account the interferometric visibility data despite working with more computationally manageable image domain data products.

We begin from the source extraction model comprising multiple point–sources proposed by Tan (1986, Chap. 1). The field of view is taken to contain N point sources, with unknown positions $\{\mathbf{x}_j\} = \{(l_j, m_j)\}, j = 1, 2, ..., N$ and fluxes F_j. Hence the desired model of the sky flux density $F(\mathbf{x})$ at position vector $\mathbf{x}$ can be written as

$$F(\mathbf{x}) = \sum_{j=1}^{N} F_j(\mathbf{x} - \mathbf{x}_j) \equiv \mathbf{F} \tag{2.2}$$

According to the Fourier transform relation described in Equation 1.3, the corresponding visibility model $\widehat{V}_k$ from $\mathbf{F}$ for a certain $\boldsymbol{u}_k = (u_k, v_k)$ can be derived as

$$\widehat{V}_k = \sum_{j=1}^{N} F_j \exp(-i2\pi\mathbf{u}_k \cdot \mathbf{x}_j) \tag{2.3}$$

Supposing that there is Gaussian noise in the visibility data $\{V_k\}$ with variance σ_k^2, the probability of measuring $\{V_k\}$ given the model $\widehat{V}_k$ with unknown parameters $\{\mathbf{x}_j\}$ and $\{F_j\}$, which will be the likelihood in Bayes' theorem, is

$$\begin{aligned} P(\{V_k\}|\{F_j\}, \{\mathbf{x}_j\}) &= \prod_{k=1}^{M} \frac{1}{2\pi\sigma_k^2} \exp\left(\frac{-|V_k - \widehat{V}_k|^2}{2\sigma_k^2}\right) \\ &= \exp\left(-\chi^2/2\right) \prod_{k=1}^{M} \frac{1}{2\pi\sigma_k^2} \end{aligned} \tag{2.4}$$

where χ^2 is given by

$$\chi^2 = \sum_{k=1}^{M} \frac{|V_k - \widehat{V}_k|^2}{\sigma_k^2} \tag{2.5}$$

and can be further calculated as:

$$\chi^2 = \sum_{k=1}^{M} \frac{|V_k - \sum_{j=1}^{N} F_j \exp(-i2\pi \mathbf{u}_k \cdot \mathbf{x}_j)|^2}{\sigma_k^2}$$
$$= \sum_{k=1}^{M} \frac{|V_k|^2}{\sigma_k^2} - 2\sum_{j=1}^{N} F_j \Re\Big[\sum_{k=1}^{M} \frac{V_k \exp(-i2\pi \mathbf{u}_k \cdot \mathbf{x}_j)}{\sigma_k^2}\Big] + \sum_{j_1=1}^{N}\sum_{j_2=1}^{N} F_{j_1} F_{j_2} \Re\Big[\sum_{k=1}^{M} \frac{\exp[-i2\pi \mathbf{u}_k \cdot (\mathbf{x}_{j_1} - \mathbf{x}_{j_2})]}{\sigma_k^2}\Big]$$

According to the definition of natural weighted dirty image and dirty beam in Equation 1.11 and 1.12, χ^2 can be written as

$$\chi^2 = \sum_{k=1}^{M} \frac{|V_k|^2}{\sigma_k^2} - \frac{1}{\sigma^2}(2\boldsymbol{D}^{\mathbf{T}}\mathbf{F} - \mathbf{F}^{\mathbf{T}}\mathbf{B}\mathbf{F}) \tag{2.6}$$

where $I_D(\boldsymbol{x}_j) \equiv \boldsymbol{D}$ denotes values of the dirty map I_D at position $\boldsymbol{x_j}$, and $B(\boldsymbol{x}_{j_1} - \boldsymbol{x}_{j_2}) \equiv \boldsymbol{B}$ denotes the corresponding dirty beam values. To cover all dirty beam values corresponding to any position $(\boldsymbol{x}_{j_1} - \boldsymbol{x}_{j_2})$, we make the dirty beam twice as the size of the dirty image.

The dirty image and dirty beam are involved in calculating the likelihood $P(\{V_k\}|\{F_j\}, \{\mathbf{x}_j\})$. According to Bayes' theorem in Equation 2.1, the posterior probability $P(\{F_j\}, \{\mathbf{x}_j\}|\{V_k\})$ can be written as:

$$P(\{F_j\}, \{\mathbf{x}_j\}|\{V_k\}) = \frac{\exp\Big(-\sum_j \frac{|V_k|^2}{2\sigma_k^2} + \frac{1}{\sigma^2}(\boldsymbol{D}^{\mathbf{T}}\mathbf{F} - \frac{1}{2}\mathbf{F}^{\mathbf{T}}\mathbf{B}\mathbf{F})\Big) P(\{F_j\}, \{\mathbf{x}_j\})}{P(\{V_k\})} \tag{2.7}$$

where $P(\{F_j\}, \{\mathbf{x}_j\})$ is the prior probability representing our prior knowledge about $\{F_j\}$ and $\{\mathbf{x}_j\}$. For simplicity, we denote the likelihood as $\mathcal{L}$.

Tan (1986) explained the difficulties in finding analytical solutions to this Bayesian model when $N > 2$. Since our model involves a high–dimensional parameter space (N fluxes, $2N$ position coordinates and one source number), for which the probability distribution is difficult to sample directly, we adopt the MCMC process to estimate the target probability distribution, so as to determine the most likely parameter values in our model.

2.2 Markov chain monte carlo

MCMC is a method of approximating high dimensional probability distributions through a random walk. It utilises computational power to generate a sequence of samples from the target probability distribution. This sequence of samples is called a "Markov chain", in which

the current sample at time t ($t \geq 1$) depends only on the sample at time $(t-1)$, but not on any previous samples (Gilks et al. 1995). When this generated Markov chain converges to its equilibrium distribution, the density of these samples in the parameter space is proportional to the target posterior probability (Hastings 1970).

We now outline how to approximate the likelihood $P(I|\boldsymbol{\theta})$ through MCMC. An arbitrary parameter $\boldsymbol{\theta}_t$ is chosen as the first sample, then a new sample candidate $\boldsymbol{\theta}'$ is generated by a chosen function of the current sample $\boldsymbol{\theta}_t$. In the Metropolis–Hastings algorithm, this function is typically a Gaussian function centred on $\boldsymbol{\theta}_t$. The Metropolis–Hastings algorithm is a widely–used MCMC method. It was first proposed by Metropolis et al. (1953) and generalised by Hastings (1970).

Next, $\boldsymbol{\theta}'$ is either rejected or accepted as the next sample $\boldsymbol{\theta}_{t+1}$ according to the 'acceptance ratio'. The acceptance ratio used here is defined as

$$A(\boldsymbol{\theta}', \boldsymbol{\theta}_t) = \min(1, \frac{P(I|\boldsymbol{\theta}')}{P(I|\boldsymbol{\theta}_t)}) \tag{2.8}$$

A random number will be generated from a uniform distribution within the range [0, 1]. If the number is no larger than the acceptance ratio then $\boldsymbol{\theta}'$ will be taken as $\boldsymbol{\theta}_{t+1}$ as the accepted new sample; otherwise, $\boldsymbol{\theta}_{t+1}$ will remain at $\boldsymbol{\theta}_t$.

The MCMC sample selection criteria can therefore be written as:

$$\boldsymbol{\theta}_{t+1} = \begin{cases} \boldsymbol{\theta}', \text{if } A(\boldsymbol{\theta}', \boldsymbol{\theta}_t) \geq \mathit{rand} \\ \boldsymbol{\theta}_t, \text{otherwise} \end{cases} \tag{2.9}$$

where *rand* is the random number generated between 0 and 1.

This process of proposing and selecting samples is repeated through iterations, so as to generate a chain of samples. The distribution of these samples is then used to approximate the target likelihood.

The MCMC driver used in implementing our Bayesian model is `BayeSys` (Skilling 2004). It is based on the Metropolis–Hastings algorithm, with improvements in efficiency and the use of atoms (Hobson & McLachlan 2003). In each iteration of `BayeSys`, multiple models will be searched, but only one of these will be selected as the output. Since the source number is unknown, we do not fix the number of atoms n; instead, n is treated as a parameter and is dynamically adjusted by proposing a model with one additional atom or one fewer when using `BayeSys`. The adjustment intervals are sampled randomly from the prior probability $P(n)$. We use a Poisson prior with $\alpha = 1$:

$$P(n) = (n!e)^{-1} \tag{2.10}$$

Further details of `BayeSys` usage and control of the atom number are set out in Skilling (2004).

In our application, each atom represents a sample $\boldsymbol{\theta}_j = \{a_j, F_j, l_j, m_j\}$, respectively specifying the atom index, flux and location information of the detected point source. Thus, the atom number can be regarded as the number of point sources found in a certain model. We denote the atoms found in the same model with the same atom index number by a_j, so that we can readily distinguish atoms found in different iterations. We shall later choose pixel numbers to describe positions, rather than angular units; this is consistent with the procedure of taking pixel values from dirty images and dirty beams for each model. It also facilitates the application of clustering algorithms and the choice of algorithm parameters.

At iteration $i + 1$, the likelihood can be calculated by applying the set of atoms following Equation 2.4, as

$$\mathcal{L}_{i+1} = \exp\left(-\chi^2_{i+1}/2\right) \prod_{k=1}^{M} \frac{1}{2\pi\sigma_k^2} \tag{2.11}$$

The logarithm of the ratio of likelihood at iteration $i + 1$ and i then becomes

$$\log \frac{\mathcal{L}_{i+1}}{\mathcal{L}_i} = (\chi^2_i - \chi^2_{i+1})/2 \tag{2.12}$$

In this way, the first term of Equation 2.6, involving visibility data, cancels out. The full dirty image and beam is pre–calculated, and only the values at the locations of the proposed atoms are taken for each model. Other than dirty images and beams, the primary beams are also needed to correct the attenuated source fluxes on the image plane, improving the accuracy of the flux information. This can be done by means of a division of the primary beam from the dirty image at the beginning; the primary beam corrected dirty image is then used as the `BayeSys` input.

An earlier MCMC approach in radio interferometry by Lochner et al. (2015) works directly on visibility data. If M is the visibility number and N is the source number then it requires $M \times N$ operations for each iteration. M is usually much larger than N. A total of 55 CPU hours was needed characterise 17 sources. By introducing the equivalent expression χ^2, the number of operations is reduced to N^2.

In `BayeSys`, parameters are set in the range $(0, 1)$. In the present application we write the flux variable F as

$$F = F_0 \frac{r}{1 - r} \tag{2.13}$$

where r is in the range $[0, 1)$, with a flat prior, and F_0 is a scaling constant. Thus, F ranging from zero to infinity can be represented by r. We set F_0 equal to the noise level of the dirty map as default, which usually gives good results.

We also normalise the position vector to $[0, 1)$ with a flat prior. By scaling it to the image size, it can represent our target position information (x, y) in units of pixel number. Both x and y may be decimals.

Our target distribution is the posterior distribution of the parameter space, and BayeSys has the ability to calculate the corresponding evidence and priors. Hence we need only input Equation 2.12 to BayeSys. At each iteration, BayeSys provides randomly generated rs and normalised position parameters in the range $[0, 1)$, which can be calculated as corresponding flux Fs and positions (x, y)s in pixel numbers so as to form the model. This model is then fed into Equation 2.12. Along with the calculated evidence and prior, BayeSys is able to provide models that follow the desired posterior distribution.

In our application, the final output of BayeSys typically gives rise to fewer than 200 models, although a much larger number of models will have been explored by BayeSys without being included. The output is a chain of sets of atoms. Each set of atoms represent a model and contains the atom index, positions and fluxes. The atoms can be scattered on the image plane according to their location and then clustered via an appropriate clustering algorithm. After this, source properties can be extracted from each cluster of atoms.

2.3 Clustering

The number of atoms found by BayeSys varies from model to model. This prevents us from performing a simple average of all models so as to approximate the individual source fluxes and positions. The density of atoms is proportional to the posterior distribution, so that if atoms were clustered then the clusters would represent the most likely locations of the sources, and the cluster number would indicate the most likely source number. Therefore we must first examine the atom positions and select an appropriate clustering method to find the most dense areas of clusters.

Various clustering algorithms exist with differing advantages and disadvantages. Berkhin (2006) gives an overview of clustering techniques, and introduces two main categories: hierarchical methods and partitioning methods. Each of these algorithms works under specified conditions. For example, the k–means clustering algorithm, a widely used partitioning method, requires the cluster number k as an input. The algorithm iterates to find k centroids and assigns data to them so as to form corresponding k clusters until the sum of the within–cluster variance is minimised (Lloyd 1982). But k–means is unable to work unless the cluster number is specified. Hardly any clustering algorithm works well on every type of dataset, in fact. It is necessary to select the clustering algorithm based on the data, then choose the suitable parameters to use.

In the present application, we have little information about the source number, so we cannot use algorithms such as k–means. Furthermore, the input parameters of the chosen clustering method should be easily selected with little human interference, so that our source extraction software BaSC performs batch processing with minimal supervision.

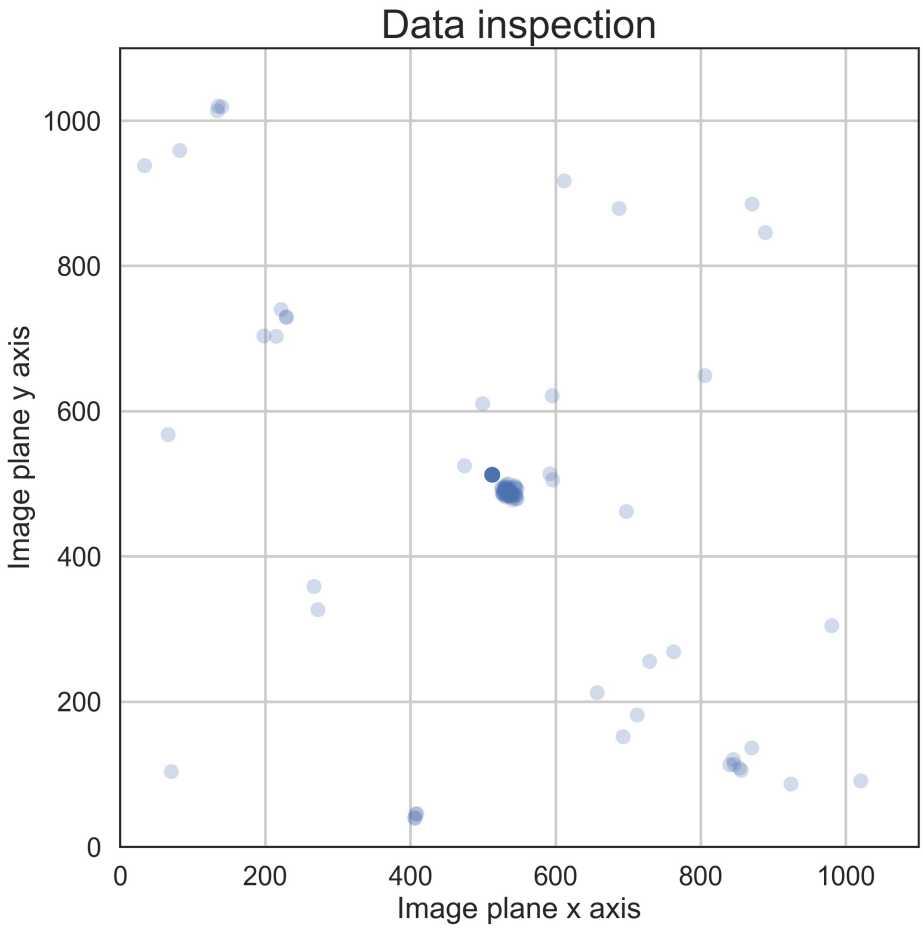


Figure 2.1: Data inspection of the BayeSys output of a two point–source simulated dataset. Blue dots on the plot represent atoms. The x–axis and y–axis show positions in units of pixel number. The blue dots are translucent so that the density of atoms is distinguished by the shade. We observe two dense clusters in the middle, and outliers scattered sparsely across the image plane.

To find the appropriate clustering method, we must first examine our data. If we simulate a two point–source data and plot its BayeSys output locations on a two–dimensional image plane, blue dots representing atoms can be seen in Figure 2.1. The blue dots are translucent so that the density of atoms can be distinguished by the shade. We observe two dense clusters in the middle, with outliers scattered sparsely across the image plane.

This simulation is generated by CASA based on observations of NGC 1808 detailed in Table 2 in Salak et al. (2016). ALMA antenna configuration 2.6 was used over an interval of 698 seconds. The two point sources were simulated with a separation of 30 pixels and an angle of

2.29 rad: one is located at the the pixel centre (512.5, 512.5) and the other at (534.5, 492.5). Their fluxes are respectively 2.24×10^{-1}Jy and 5.6×10^{-3}Jy, and the former source is 40 times brighter. The RMS of the image noise is $\sigma = 5.6 \times 10^{-4}$ Jy/beam. The resulting dirty image is 1024 by 1024 with a pixel size of 0.01 arcsec. In the following discussion we plot the positions of atoms in units of image pixel number, usually on a 1100×1100 plane to show clearly those atoms found at the image edge.

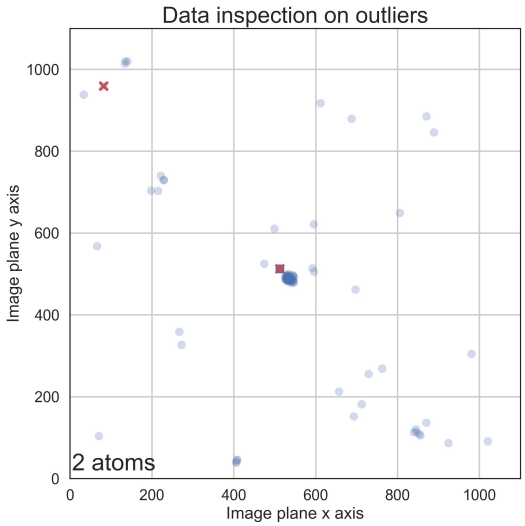
Following the testing of multiple simulations, we quickly learn that the output of `BayeSys` from different simulated data commonly contain outliers as shown in Figure 2.1, scattered loosely across the image plane. This is primarily due to the nature of MCMC and the varying number of atoms found by `BayeSys` in each model. It is emphasised that, for a certain model proposed from the stated two point–source simulated data, `BayeSys` can find a set of atoms whose parameters maximise the posterior probability distribution. The number of atoms may be one, two, or more than two.

In the example shown in Figure 2.1, either one atom is found or two. We plot atoms found in particular models in Figure 2.2, marked by red crosses. In some cases only one atom is found; the bottom right plot is an example. For models comprising two atoms, the two atoms are not necessarily located within the two most dense clusters, and one is therefore identified as an outlier.

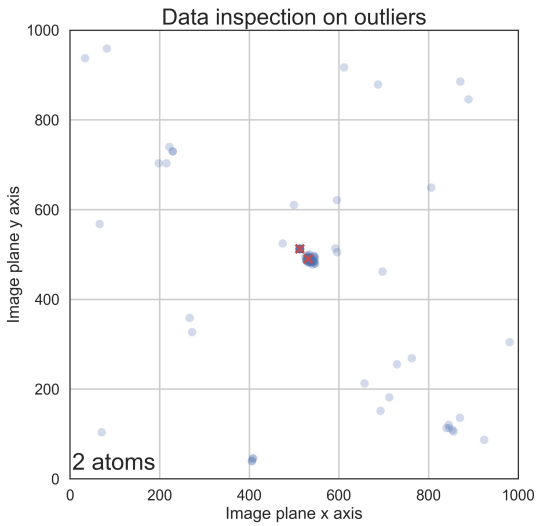
In these four cases the position of the brighter source was found within the same pixel every time, but this is not true for the dimmer source. Upon looking at the two clusters, it is clear that the cluster which indicates the location of the brighter source is far more compact than the other. This indicates that it is easier for `BayeSys` to find the brighter source than the dimmer one. Further experiments and explanations can be found in §2.5.4.

Our interest is with the dense areas of atoms, so that we must find a robust clustering algorithm that disregards outliers. We began by testing clustering methods which do not require the cluster number to be pre–specified, and soon found that Density–based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al. 1996) is a good candidate. This does not require the cluster number as input, and it also has an outlier–detection feature. We executed this algorithm via a clustering toolkit called `scikit-learn` (Pedregosa et al. 2011) in `Python`. The results are shown in Figure 2.3. The parameters used are explained in §2.3.1. Blue dots and green dots represent two different clusters found by DBSCAN. The grey dots are outliers detected by the algorithm. The x– and y–axes show positions in units of pixel number. It is apparent that DBSCAN detects outliers and finds clusters at the same time.

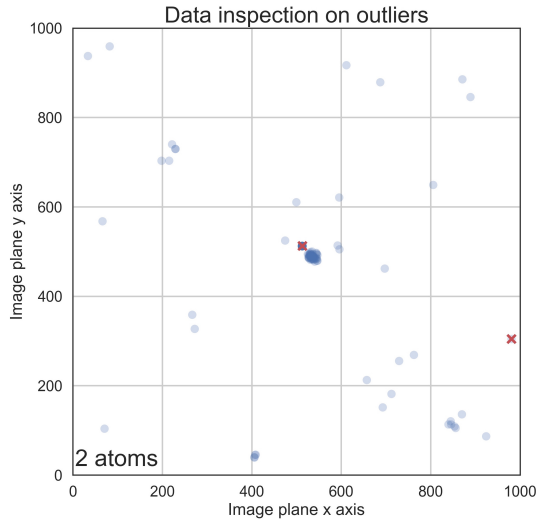
Hierarchical clustering does not require the cluster number, but algorithms such as affinity propagation (Frey & Dueck 2007) classifies even outliers into clusters. We also used the Gaussian Mixture Model (GMM) via the Expectation–Maximization(EM) algorithm (Fraley &



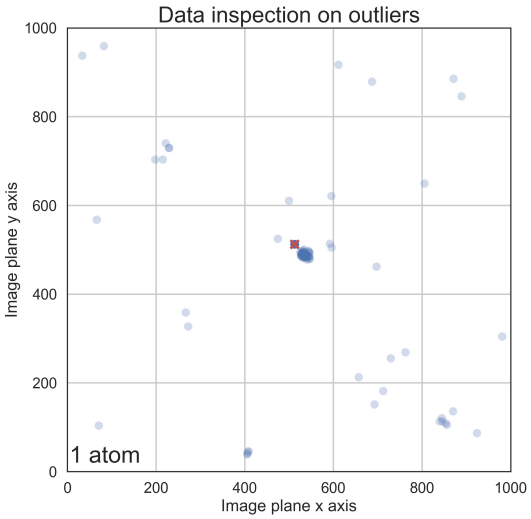
(a) Two atoms found with parameter values (x, y, F) as $(512.5, 512.7, 2.53 \times 10^{-1})$ and $(81.9, 959.2, 7.64 \times 10^{-5})$



(b) Two atoms found with parameter values (x, y, F) as $(530.9, 490.3, 4.49 \times 10^{-3})$ and $(512.3, 512.8, 2.51 \times 10^{-1})$



(c) Two atoms found have parameter values (x, y, F) as $(980.5, 304.5, 2.06 \times 10^{-4})$ and $(512.8, 512.6, 2.53 \times 10^{-1})$



(d) One atom found has parameter values (x, y, F) as $(512.7, 512.2, 2.52 \times 10^{-1})$

Figure 2.2: Demonstration of four different models found by BayeSys. The blue dots in each plot represent locations of the full output data, and the red crosses show the locations of the atoms in the model proposed. Apart from the bottom right plot, the models have two atoms.In the three two-atom models, the atoms are obviously not invariably located within the two most dense clusters.

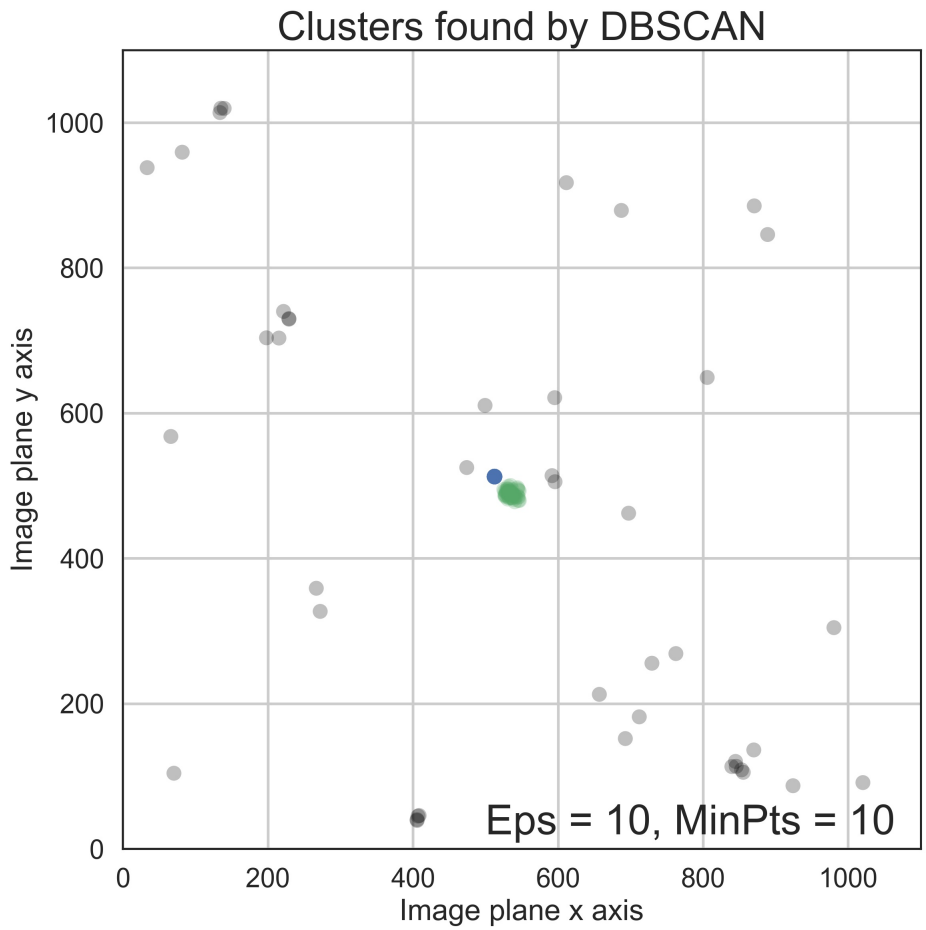


Figure 2.3: Clustering results using DBSCAN for the two point–source simulated data. Blue dots and green dots represent two distinct clusters found by DBSCAN. The grey dots are outliers detected.

Raftery 1998) to fit Gaussian models to clusters of the dataset. This does not work well on our dataset, as it made clusters of the outliers.

2.3.1 DBSCAN

We now outline how DBSCAN works with our `BayeSys` output. DBSCAN detects dense areas of points, and groups them by restricting the maximum neighbouring distance and the minimum grouping number. The maximum neighbouring distance, denoted by `Eps`, is the maximum distance between two data points which may be regarded as neighbours, and therefore belong to the same cluster. DBSCAN finds neighbours of a particular point within a radius of `Eps` and

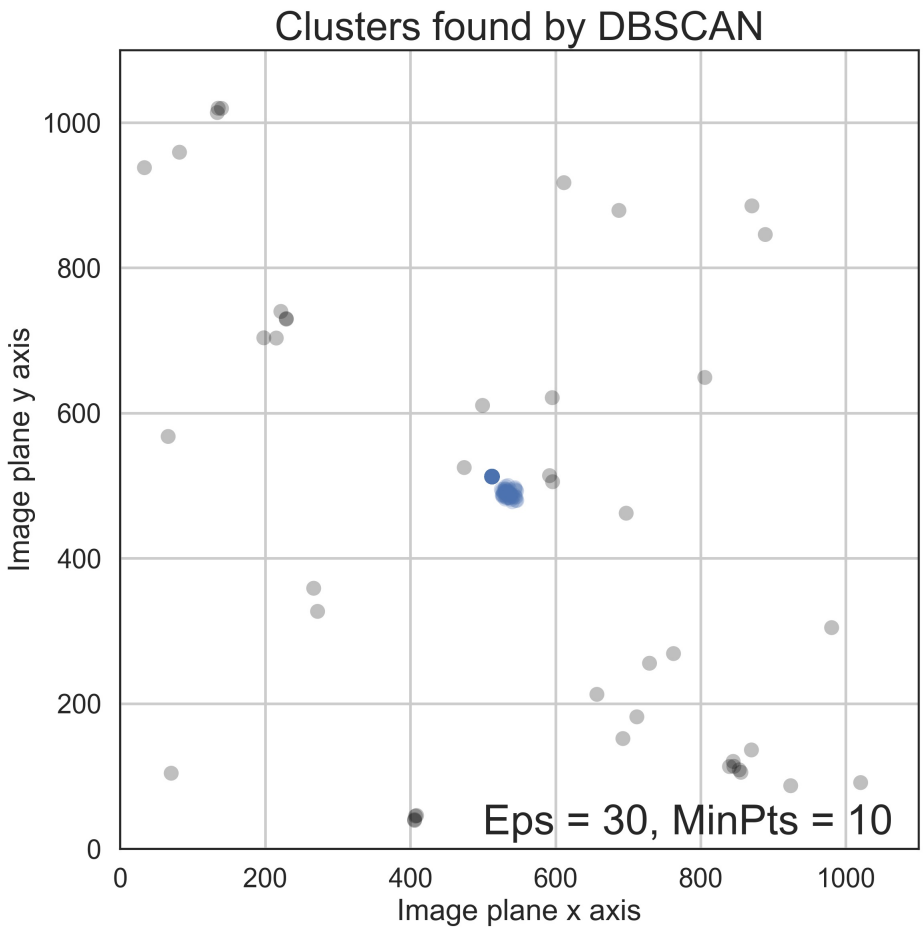


Figure 2.4: DBSCAN finds only one cluster with `Eps` = 30 and `MinPts` = 10. The blue dots represent the only cluster found by DBSCAN. The grey dots are outliers detected.

groups them together. The other parameter required by DBSCAN is the minimum grouping number `MinPts`. If a neighbourhood of points found has no less than `MinPts` points, it will be considered a cluster. These two parameters ensure that any clusters found are above a specified density. Points not assigned to any clusters are considered as outliers. Distance refers in our application to Euclidean distance.

The steps of the DBSCAN algorithm used to process `BayeSys` output are as follows (Ester et al. 1996; Schubert et al. 2017):

1) Pick a random data point which has not been assigned to a cluster or labeled as an outlier. If this point has no fewer than `MinPts` neighbours within a radius of `Eps`, it is defined as a 'core point'; otherwise it is labeled an outlier.

2) Find data points within a radius `Eps` of the core point so as to form a cluster, and then expand the cluster by adding data points within the same radius of every cluster point, including ones that are to be added. If an outlier is added to a cluster, remove its 'outlier' label.
3) Repeat the preceding two steps until all points are assigned to clusters or considered as outliers.

In each cluster, or the most dense areas of atoms detected by DBSCAN, `BayeSys` considers there to be a peak in the target probability distribution. Hence the source location is calculated as the average of the points' location within this cluster, while the outliers are removed to avoid distortion of the data.

The choice of `MinPts` and `Eps` has a considerable impact on the performance of DBSCAN, however. First, if the radius of the cluster searching area `Eps` is chosen to be too large, two nearby clusters, or two sources in our problem, would be considered as one. If `Eps` is chosen too small, conversely, outliers would be regarded as sources. Secondly, if the minimum grouping number `MinPts` is chosen too small then outliers would be considered as sources, whereas if the value is too large then even sources would be considered as outliers.

In Figure 2.3 we used the values `Eps` = 10 and `MinPts` = 10. If `Eps` is increased to 30 while `MinPts` = 10, the two clusters found earlier are considered to be one, as shown in Figure 2.4. The blue dots represent the only cluster found by DBSCAN and grey dots are outliers detected. This is because the distance between the two sets of points is no more than `Eps`, so that they are merged (Ester et al. 1996); the distance between two groups is defined as the minimum distance between any two points in each group.

If on the other had we maintain `Eps` = 10 but increase `MinPts` to 40, one of the two clusters found before is no longer considered to be a cluster. In Figure 2.5 the blue dots represent the only cluster found by DBSCAN. The grey dots are outliers detected. The increase in `MinPts` causes the less dense group of points not to be defined as a cluster but merely a set of outliers.

In the present application, `Eps` should be no less than 0.5, or half of the pixel size, to prevent atoms in the same pixel being separated into more than one cluster, and `MinPts` should be not less than 10 so that outliers are not considered as clusters. Users may choose `Eps` based on their data.

In `BaSC` we also provide a solution that will assist in batch processing by recommending a value of `Eps` for each dataset with `MinPts` = 10 fixed. The recommendation was made based on the sorted k–dist graph (Ester et al. 1996); see Figure 2.6. This figure plots the distances from k nearest neighbours for all data points in ascending order. In Figure 2.6 we set $k = 10$; the x–axis is the index number of the sorted distances of all data points from their k–nearest neighbours, and the y–axis represents the distance values. The sorted k–dist graph is used to

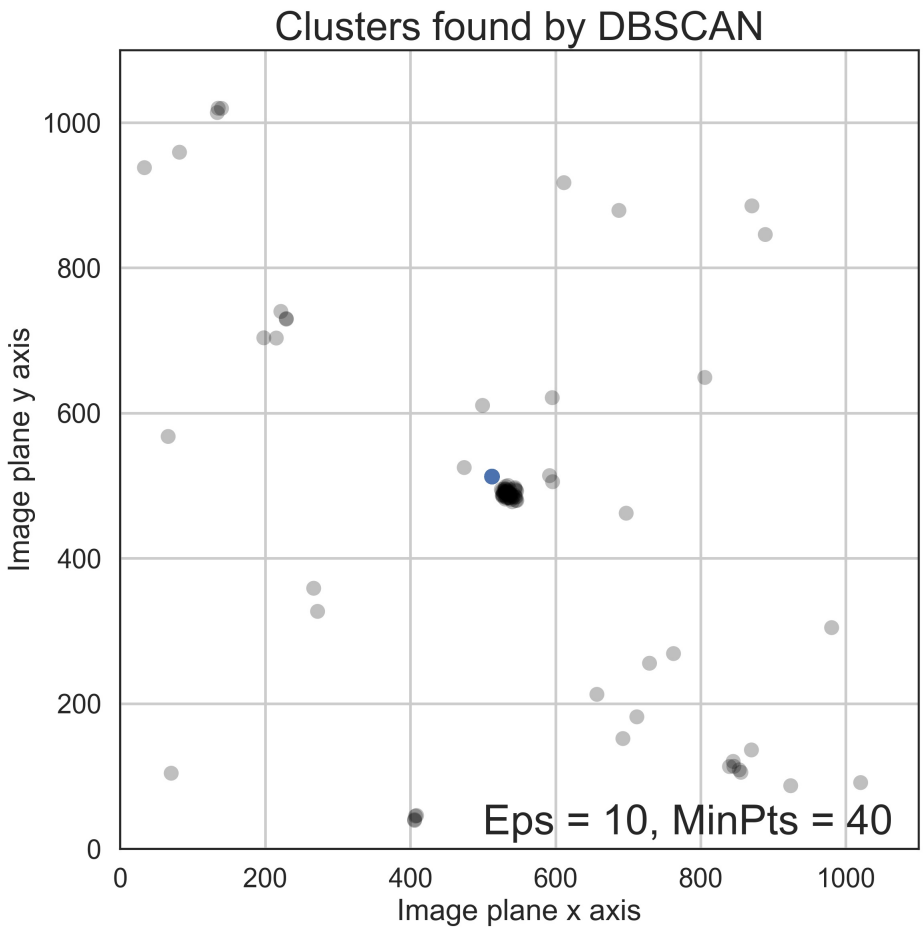


Figure 2.5: DBSCAN with `Eps` = 10 and `MinPts` = 40 finds a single cluster. The blue dots represent this cluster. The grey dots are outliers detected.

give an approximate value of `Eps` by classifying outliers and non–outliers using a threshold; see the broken line in Figure 2.6. This threshold will be the recommended `Eps`. Ester et al. (1996) points out that the threshold should be the first point of the first 'valley' at the k–dist graph shown, which is usually not easy to detect. `BaSC` users can use the recommended `Eps` directly based on our threshold searching program detailed in Appendix A, or choose `Eps` intuitively.

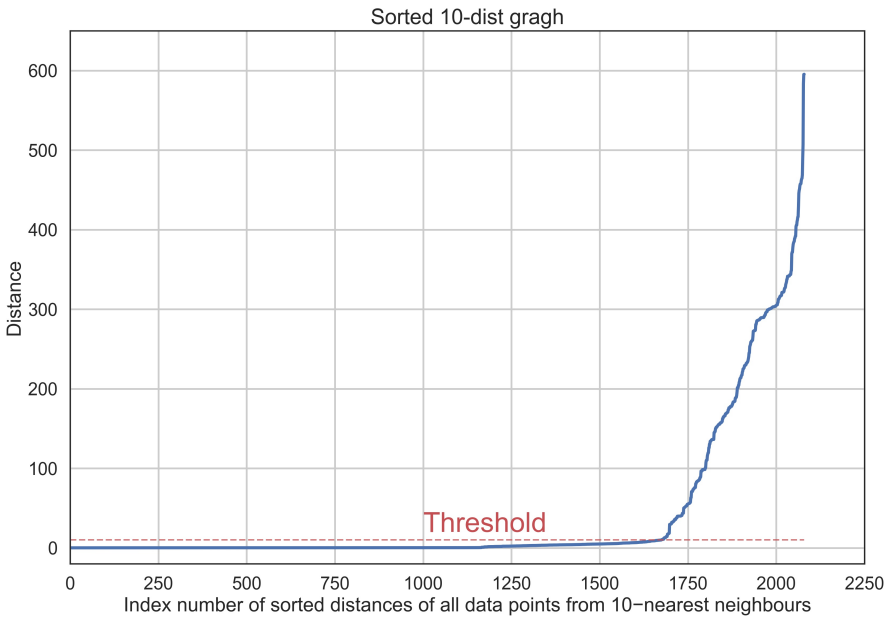


Figure 2.6: This sorted k–dist graph is used to determine an approximate value of `Eps` by classifying outliers and non–outliers using a threshold; see the broken line. The x–axis is the index number of the distances of all data points from their 10 nearest neighbours in ascending order. The y–axis represents the distance values. The threshold corresponds to the first point of the first 'valley' at the k–dist graph and is the recommended `Eps`. It is not easy to detect in this case, however.

2.3.2 OPTICS

Ordering Points To Identify the Clustering Structure (OPTICS) (Ankerst et al. 1999) is a clustering method based on a similar concept to DBSCAN without requiring the parameter `Eps`. OPTICS still asks for the parameter `MinPts`, which defines the minimum data point number required to form a cluster. In OPTICS, for a particular data point, its 'core distance' is defined as its $(\texttt{MinPts} - 1)$th shortest distance from its nearest $(\texttt{MinPts} - 1)$ data points. OPTICS also defines the 'reachability distance' for each data pair. The reachability distance of a point A to point B is the larger value of A's core distance and the distance between A and B, and conversely for the reachability distance of point B to point A.

OPTICS linearly reorders all data points by making points which are spatially close to each other into neighbours. Hence, OPTICS can produce a 'reachability–plot' in which the x–axis is the ordered data point indexes, and the y–axis is the reachability distance of the corresponding data point. The reachability–plot of the same simulated dataset is shown in Figure 2.7. We use the OPTICS package implemented in `scikit-learn` (Pedregosa et al. 2011), with $\texttt{MinPts} = 10$.

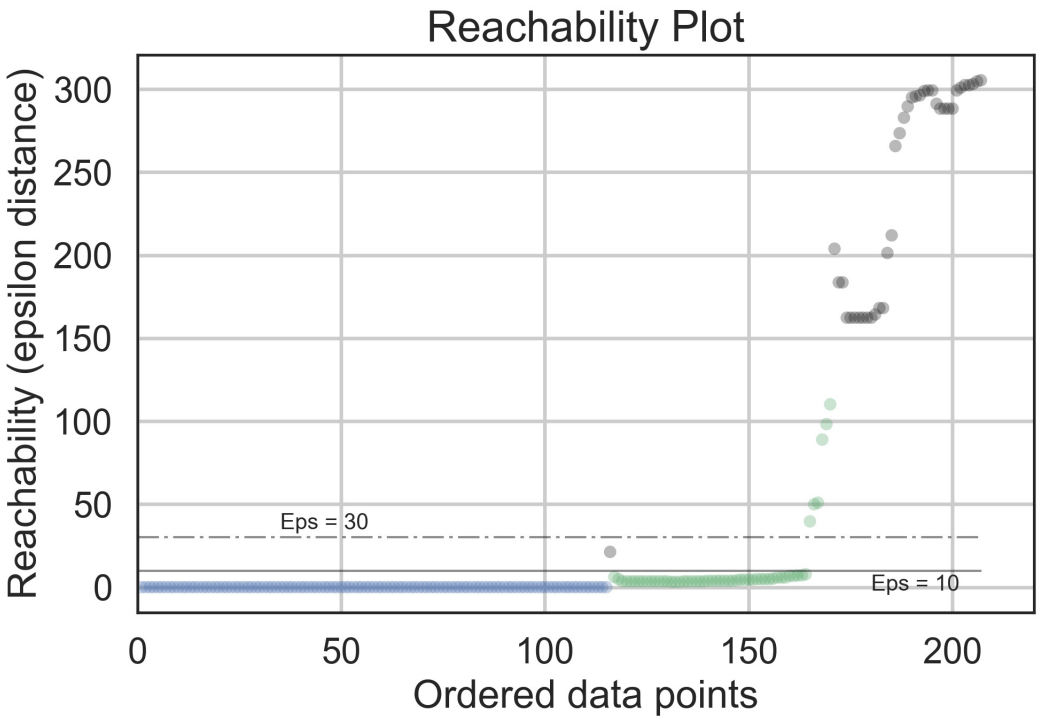


Figure 2.7: OPTICS reachability plot of the simulated two point–source data. Blue dots and green dots represent two distinct clusters categorised by OPTICS, and the grey dots are outliers. The x–axis represents the index of OPTICS sorted points, and the y–axis represents the reachability distances of the corresponding points. The grey solid and dashed lines show two different choices of reachability distance, which are equivalent to two different `Eps` choices when using DBSCAN.

The reachability plot will change if `MinPts` changes.

In Figure 2.7, blue dots and green dots represent two distinct clusters categorised by OPTICS. The grey dots are categorised as 'undefined', or outliers. The x–axis represents the points sorted by OPTICS, and the y–axis represents their reachability distances. Since the reachability distances of denser areas of points are smaller than the less dense areas, the deeper valleys represent denser clusters. For instance, the blue valley shown in Figure 2.7 represents a denser cluster than the green valley. Points above valleys are considered as points outside clusters, or in other words as outliers.

Given a particular reachability distance as threshold, OPTICS can work just like DBSCAN with a predefined `MinPts` and a chosen reachability distance, the latter of which is equivalent to `Eps` in DBSCAN. The reachability distance is therefore also known as 'epsilon distance', and OPTICS can be used to estimate the parameter `Eps` for DBSCAN.

As an example, the dashed line in Figure 2.7 sets the epsilon distance as 30, causing the

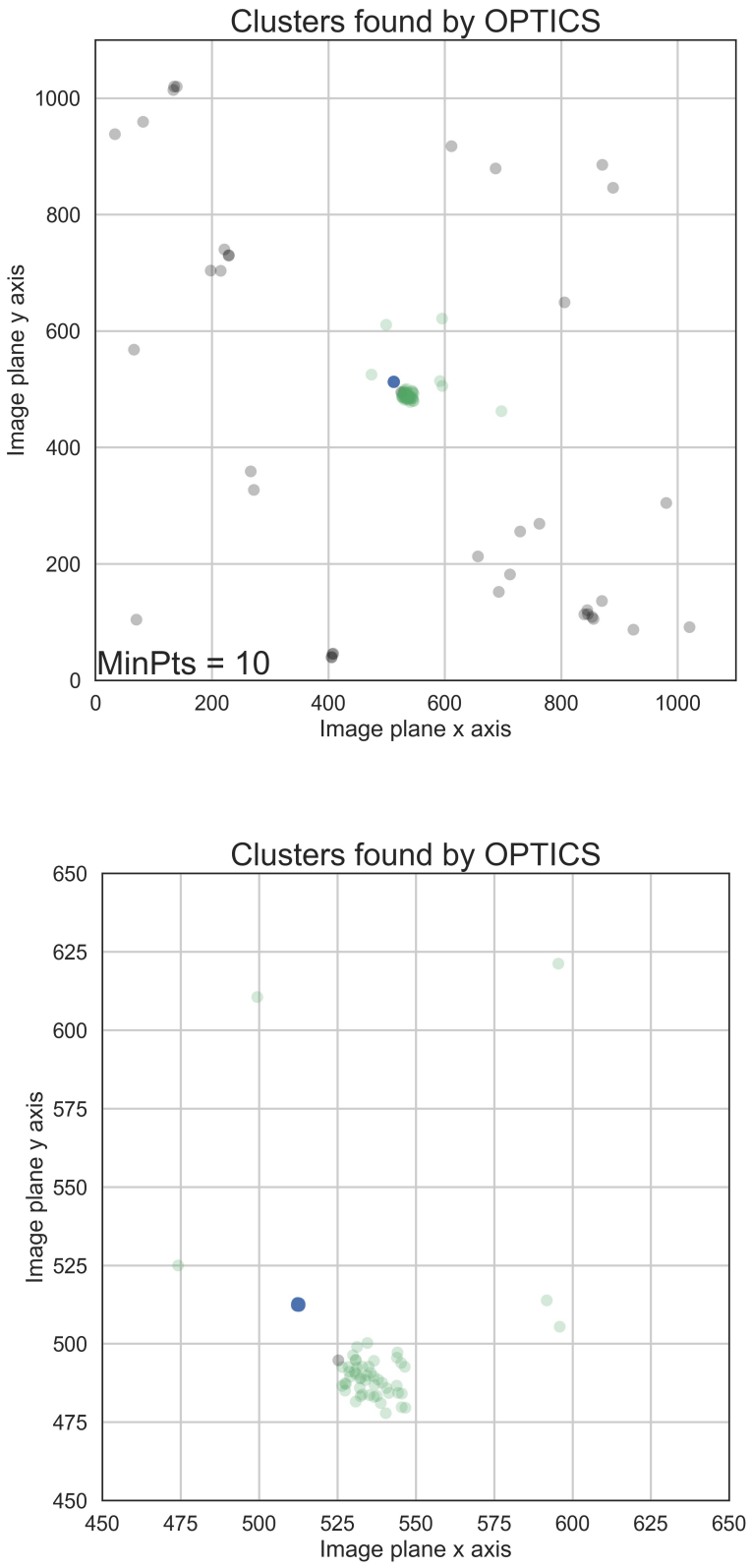


Figure 2.8: OPTICS clustering results with `MinPts` = 10. The lower figure is an enlarged version of the centre of the image in the upper figure. Blue dots and green dots represent two different clusters found by OPTICS, and the grey dots are outliers detected. The x– and y–axes show positions in units of pixel number. OPTICS mistakenly treats data that should be outliers as a part of a cluster: see the green dots scattered far from the two clusters.

two valleys to merge into one. If the epsilon distance is greater than 50, outliers would be included in clusters. The solid line sets the epsilon distance as 10 as a visually good estimate, separating the two clusters well. Thus `Eps = 10` is a good recommendation for DBSCAN. However, each dataset with different `MinPts` produces a different reachability plot, and the choice of the appropriate `Eps` requires human intervention. We do not therefore regard OPTICS as a recommended method for estimating the parameter `Eps` in DBSCAN in our package `BaSC`.

An example of an OPTICS result with `MinPts = 10` from the same simulated dataset is shown in Figure 2.8. The lower figure is an enlarged version of the centre of the image in the upper figure. Blue dots and green dots represent two different clusters found by OPTICS, and the grey dots are outliers detected. OPTICS mistakenly treats data that should be outliers as a part of a cluster; see the green dots scattered far from the two clusters. This can be corrected by choosing the right range of data points within each valley (Ankerst et al. 1999). In Figure 2.7, for example, the steep part of the green valley needs to be reduced. It can be adjusted through a parameter `rejection_ratio` in the `Python` package. The procedure is not straightforward, however, and consequently OPTICS is unsuitable for batch processing in such applications.

There is also a grey dot at the edge of the green cluster in the lower plot in Figure 2.8, which is considered as an outlier. This data point can also be found in the reachability plot in Figure 2.7, where it is shown as a grey dot between the blue and green valleys. OPTICS tends to conclude that points at the edge of a cluster are outliers, because they have large reachability distances compared to points in the central cluster.

If `MinPts` is decreased, outliers at the image corners form into clusters; see the top plot in Figure 2.9. Two more clusters consisting of outliers are found: the orange dots form a ring–like cluster, and the purple dots form a small cluster at the top left corner. If `MinPts` is increased, more outliers are assigned to clusters around the two prevailing clusters; see the lower plot in Figure 2.9.

Although OPTICS does not require the parameter `Eps`, its clustering conclusions show that it is not suitable in the present application, and it will not form part of our `BaSC` package.

2.3.3 HDBSCAN

Hierarchical Density–based Spatial Clustering of Applications with Noise (HDBSCAN) is a hierarchical clustering algorithm based on DBSCAN (Campello et al. 2013), but it does not require the parameter `Eps`. HDBSCAN defines a ‘mutual reachability distance’ which is identical to the reachability distance defined in OPTICS.

Based on the reachability distance metric, a minimum spanning tree can be constructed (Campello et al. 2015). Figure 2.10 shows a minimum spanning tree constructed by HDBSCAN based on the same simulated data. The black dots are data points connected by lines in different

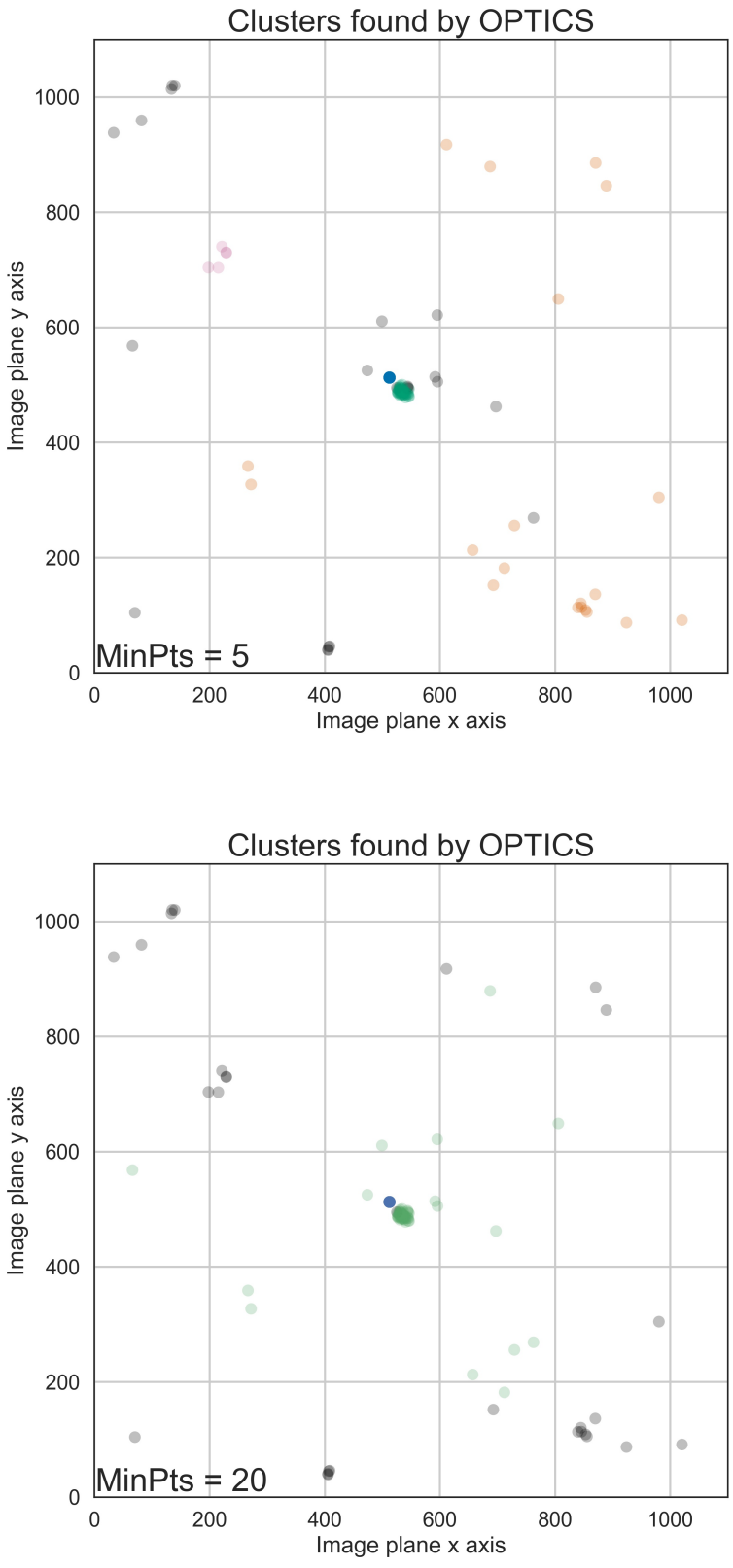


Figure 2.9: Clusters found by OPTICS with different values of `MinPts`. The grey dots are outliers detected. In the upper figure, with `MinPts` = 5, OPTICS finds outliers as clusters such as the orange dots which form a ring–like cluster, and the purple dots which form a small cluster at the top left corner. In the bottom figure, with `MinPts` = 20, OPTICS erroneously treats more outliers as a part of a cluster; see the green dots scattered around the two clusters across the image plane. The x– and y–axes show positions in units of pixel number.

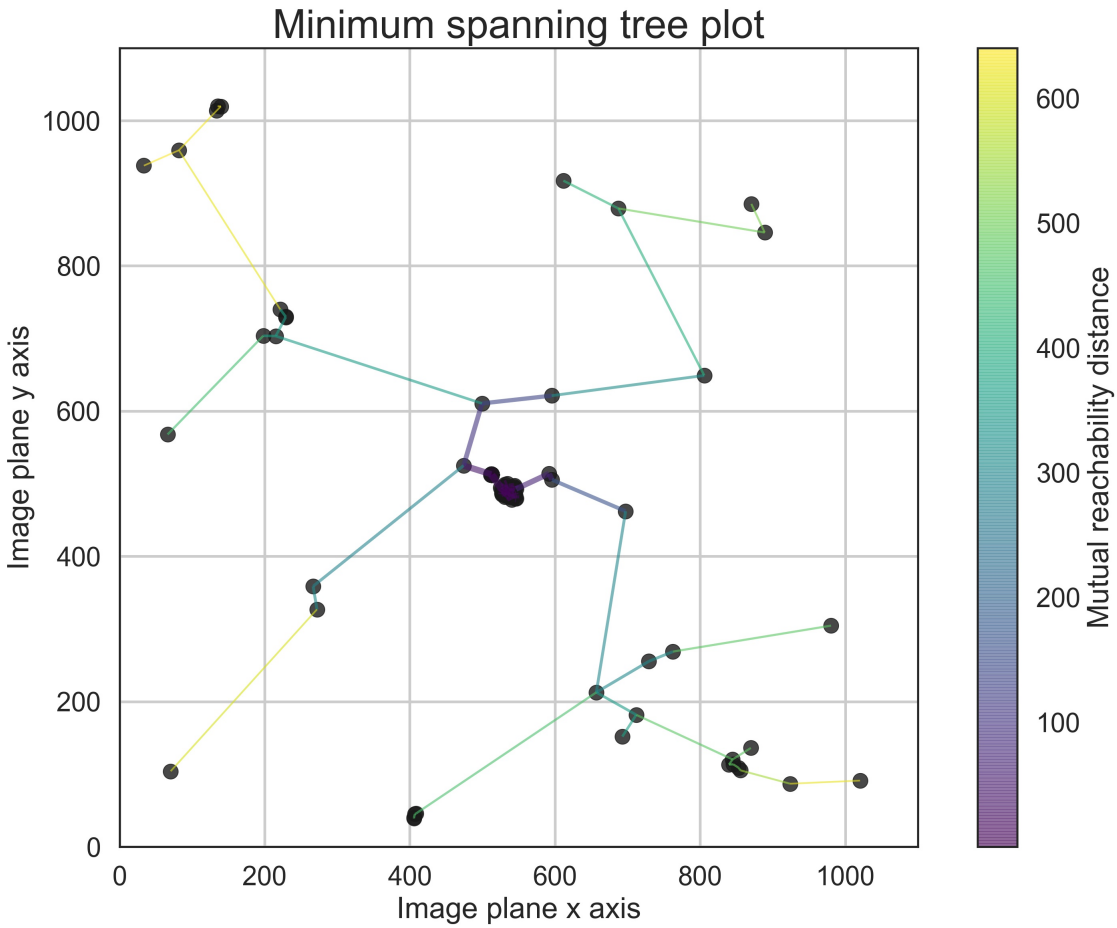


Figure 2.10: This is a minimum spanning tree plot. Black dots are data points connected by lines in different colours. The colour of the line represents the value of the mutual reachability distance. Points connected by lines of the same colour are mutually reachable. The x– and y–axes show positions in units of pixel number.

colours. The colour of the line represents the value of the mutual reachability distance: the darker the colour is, the greater the mutual reachability distance, and the higher the data point density. Points connected by the same colour are mutually reachable. The HDBSCAN `Python` library (McInnes et al. 2017) used is based on Campello et al. (2013); McInnes & Healy (2017). It can be downloaded and installed using the instructions on website `https://pypi.org/project/hdbscan/`.

The tree can be converted into a hierarchy of connected data points, and clusters can then be extracted. Campello et al. (2015) also proposed an outlier detection method for HDBSCAN, named GLOSH, an acronym of Global–Local Outlier Scores from Hierarchies. As a result

HDBSCAN has the capability to work with our data.

The only parameter that must be entered into the HDBSCAN package by hand is `MinPts`, which represents the minimum cluster size. There is a further parameter `MinSamples` in the package used; its default value is equal to `MinPts`. This parameter controls the sensitivity of HDBSCAN in picking up less dense clusters. With a larger value of `min_samples` the algorithm is more likely to ignore a number of outliers which cluster together, whereas a smaller `min_samples` may cause those outliers to be regarded as clusters. In the present application, we recommend starting with `MinPts` = 10 so as to pick up as many clusters as possible, and then set `min_samples` = 30 to prevent misidentifying lower density clusters (which are more likely to be outliers) as actual clusters.

We now present clustering results using HDBSCAN with different parameters to process our data. In the upper plot in Figure 2.11, we set `MinPts` = 10 and `min_samples` = 10. As well as the two clusters shown in blue and green, the group of outliers in purple are considered as a cluster because of the the small value of `min_samples` chosen. If `min_samples` is increased to 30, HDBSCAN correctly finds two clusters in the lower plot. The grey dots are outliers detected.

If `BaSC` is used to deal with a large number of maps with minimal human intervention, we suggest using DBSCAN with the recommended `Eps` value or HDBSCAN with `MinPts` = 10 and `min_samples` = 30. If users are concerned about the clustering results of certain maps, they are advised to reset parameter values manually.

2.4 Bayesian Resolving Power

We now turn to the theoretical resolution limit of our source extraction method, referred to below as the 'Bayesian resolving power'. In this section, we consider the ability of our method to constrain the position of a single source.

Based on the Bayesian single point–source model described in (Tan 1986, Chap. 1), if the source location is unknown and the dirty image $I_D(\mathbf{x}) \geq 3\sigma/\sqrt{M}$, the most probable source location is at the positive maximum of the dirty image. M denotes the number of visibility data. The posterior probability for the source position, given the visibility data, can be written as:

$$\mathrm{P}(\mathbf{x}|\{V_j\}) \propto \exp\left[I_D(\mathbf{x})^2/2\sigma^2\right] \text{ for } I_D(\mathbf{x}) \geq 3\sigma/\sqrt{M} \tag{2.14}$$

If the single source is at $\mathbf{x} = (0, 0)$ with a flux F_0, $I_D(\mathbf{x})$ can be written as:

$$I_D(\mathbf{x}) = F_0 B(\mathbf{x}) = F_0 \sum_{k=1}^{M} \frac{1}{\sigma_k^2} \cos(2\pi \mathbf{u}_k \cdot \mathbf{x}) \tag{2.15}$$

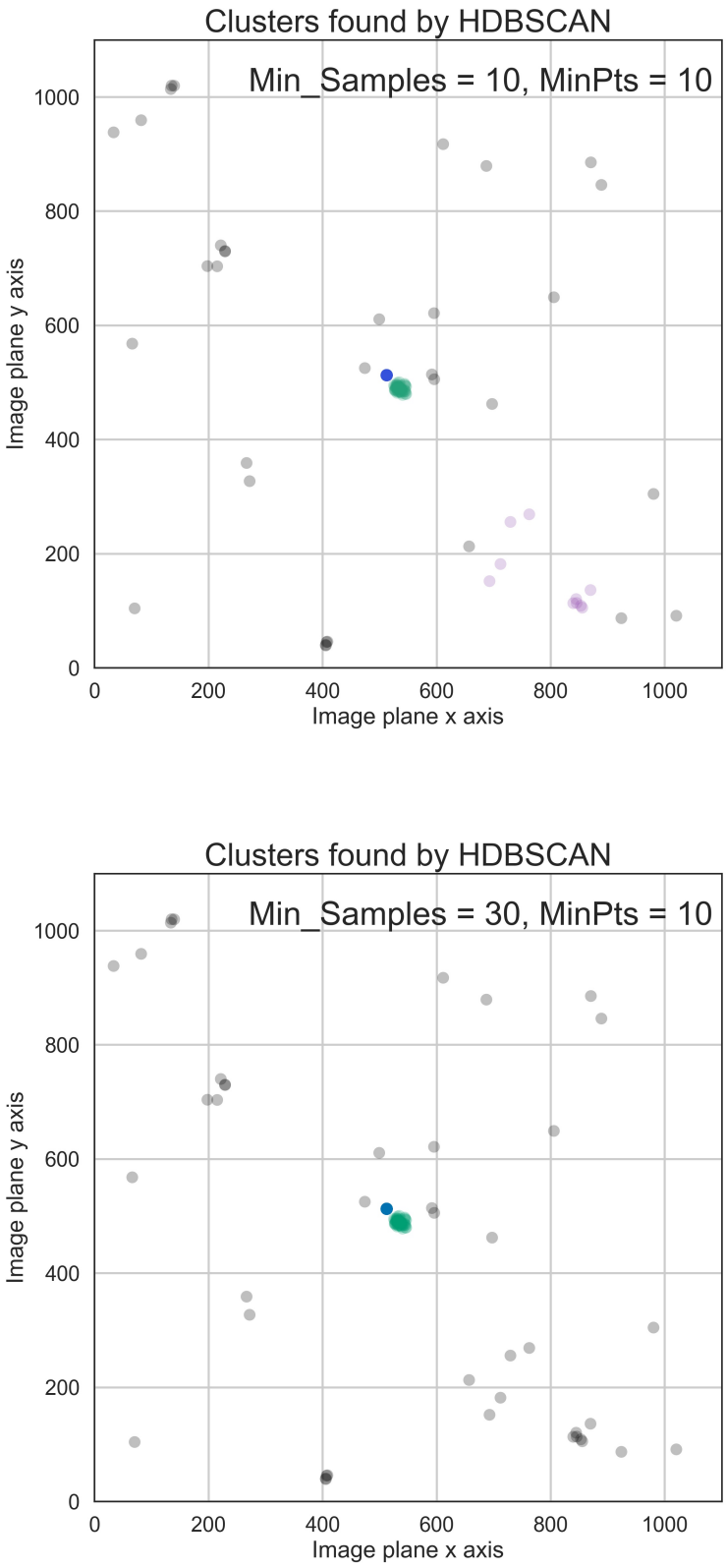


Figure 2.11: Demonstration of clustering results using HDBSCAN with different parameters on the same simulated data. In the upper plot in Figure 2.11, `MinPts` = 10 and `min_samples` = 10. As well as the two clusters in blue and green, the group of outliers in purple are considered as a cluster because of the small value of `min_samples`. If `min_samples` is increased to 30, HDBSCAN correctly finds just two clusters in the bottom plot (in blue and green). The grey dots are outliers detected.

where $B(\mathbf{x})$ is the dirty beam defined in Equation 1.12. By means of a Taylor expansion, the dirty beam an be rewritten as:

$$B(\mathbf{x}) \approx 1 + \frac{1}{2}\Big(\frac{\partial^2 B(\mathbf{x})}{\partial^2 x}\Big|_{\mathbf{x}=\mathbf{0}} + 2\frac{\partial^2 B(\mathbf{x})}{\partial x \partial y}\Big|_{\mathbf{x}=\mathbf{0}} + \frac{\partial^2 B(\mathbf{x})}{\partial^2 y}\Big|_{\mathbf{x}=\mathbf{0}}\Big) \tag{2.16}$$

We change to suffix notation here for brevity, and write $B(\mathbf{x})$ as

$$B(\mathbf{x}) \approx 1 + \frac{1}{2}\delta^2_{ij} B(\boldsymbol{x})_{|\mathbf{0}} x_i x_j = 1 + 4\pi^2\sigma^2 \sum_k \frac{(u_k)_i (u_k)_j}{\sigma_k^2} \tag{2.17}$$

From Equations 2.14, 2.15 and 2.16, we obtain the posterior probability of the source position as:

$$P(\mathbf{x}|\{V_j\}) \propto \exp\Big[-2\pi^2 F_0^2 \sum_{ijk} \frac{(u_k)_i (u_k)_j}{\sigma_k^2} x_i x_j \Big] \tag{2.18}$$

Based on the analysis of the multivariate Gaussian in Appendix B, the position covariance of the single point source is:

$$\langle \delta x_i \delta x_j \rangle = \frac{\sigma^2}{F_0^2} \Delta x_{ij}^2 \tag{2.19}$$

where Δx_{ij} is a characteristic width determined by the dirty beam defined as:

$$(\Delta x_{ij})^{-1} = \partial^2_{ij} B(\mathbf{x})_{|\mathbf{x}=\mathbf{0}} = 4\pi^2\sigma^2 \sum_k \frac{(u_k)_i (u_k)_j}{\sigma_k^2} \tag{2.20}$$

This expression describes the characteristic width of the central part of the dirty beam, for which a good approximate representation from `CASA` is an ellipse determined by a semi–major axis a, a semi–minor axis b and an angle θ.

The dimensions of the resolving ellipse can be defined by a/S and b/S, where S is the signal–to–noise ratio (SNR) defined as F_0/σ. Numerical experiments designed to test the Bayesian resolving power will be discussed in Section 2.5.

2.5 Experiments and results

This section describes numerical experiments conducted with our proposed package `BaSC`. for clustering, DBSCAN is used, with manually chosen values of `Eps` and `MinPts`. A simplified flowchart of `BaSC` is shown in Figure 2.12. In this section we first explore the resolution limit that `BaSC` can achieve, and compare it with the formula derived in Equation 2.20. We then compare the performance of `BaSC` with that of `SExtractor` on the same simulated dataset, and then make a similar comparison between `BaSC` and `Aegean`. Finally, the robustness of `BaSC` is tested on simulated data with two point sources and a wide dynamic range of fluxes.

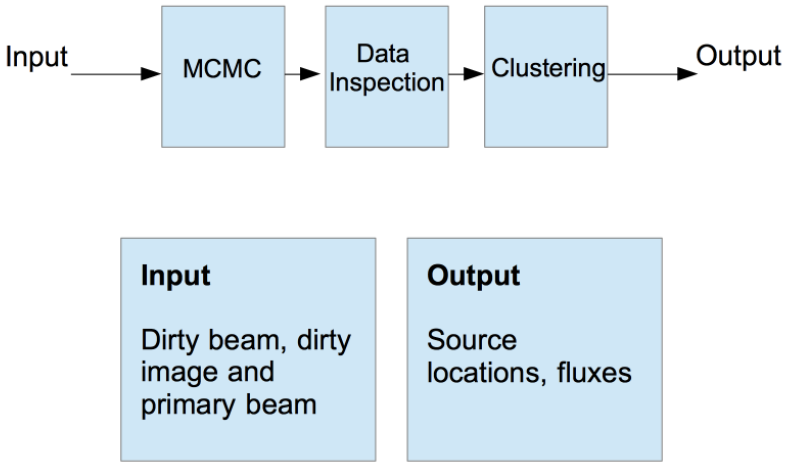


Figure 2.12: Simplified flow chart of BaSC. The input data is first processed via the MCMC driver BayeSys. After users have inspected the BayeSys output data, they will be asked to choose their preferred clustering methods and relevant parameters as necessary. The output will be the source locations expressed in pixel numbers, and fluxes in units of Jy.

2.5.1 Bayesian resolving power test

To test the formula for the Bayesian resolving power in Equation 2.20, we generated a test dataset. We made a sky model of a single point source in which every pixel of the image was zero except for the image centre, where the point source lies. The image size was chosen to be 2048 by 2048, with a cell size of 0.01 arcsec; as a result, these images are squares 20.48 arcsec across. Since the image size is substantially smaller than the corresponding primary beam, the effects of attenuated source fluxes can be neglected. We hen feed this point–source model into the simalma task in CASA with the ALMA antenna configuration 2.6. The simulated observation time was specified as 698 seconds. Other observation configurations are the same as in Salak et al. (2016).

The simalma task outputs the dirty image, dirty beam and primary beam from the point–source model. These image products are then input to BaSC so as to obtain the atom positions. We made 5 sets of simulated data with SNR equal to 10, 20, 40, 80 and 160 respectively, but with the same noise $\sigma = 5.6 \times 10^{-4}$ Jy/beam.

Table 2.1: Positional accuracy of BaSC results versus theoretical predictions.

SNR	σ_x	σ_y	Prediction
160	0.292	0.288	0.147
80	0.574	0.547	0.294
40	0.670	0.634	0.587
20	1.394	1.190	1.175
10	2.188	2.386	2.351

The results are shown in Table 2.1. Here, σ_x and σ_y represent the uncertainties in source position in x and y. These can be calculated as the standard deviation from Gaussian fitting of the atom positions (x, y)s. Prediction is made according to the formula for the resolving power. The standard deviation for each axis of the position data matches the predicted values closely if SNR =10, 20 and 40.

As the SNR increases there is a decrease in σ_x and σ_y and the Bayesian resolving power, implying less uncertainty in the source position. We also see that the point source is more easily detected when the dirty image is less noisy. However, when SNR = 80, the Bayesian resolving power falls to a value close to the uncertainty within a single pixel, which is equal to $1/\sqrt{12} \approx 0.29$ and is known in electrical engineering as 'quantisation noise'. As SNR increases to 160, σ_x and σ_y reach the single cell uncertainty limit. Although the Bayesian resolving power is even smaller, at 0.147, it is impossible to achieve.

2.5.2 Comparison between BaSC and SExtractor

To test and compare the performance of BaSC and SExtractor, we used the same simulation configurations to generate 100 dirty images with the same dirty beam and primary beam. The image noise is set to $\sigma = 5.6 \times 10^{-4}$ Jy/beam. Each simulation contains two point sources with identical fluxes but different distances and angles. One point source is fixed at the phase centre, and the other is separated by a distance that is distributed uniformly in the range $[0, 1]$ in arcsec. A uniform distribution is preferred because the final results will be binned by distance, and each bin consequently has the same number of samples. The angle of the line formed by these two points is also distributed uniformly. In this comparison the SNR is set at 10, so that the fluxes of the two point sources are both 5.6×10^{-3} Jy.

Figure 2.13 shows 16 dirty images from the 100 dirty images made for this experiment. These images are zoomed in to the central region, to highlight the location of the two sources.

The SExtractor used here is Version 2.5. We set its detection threshold to 3σ, and the minimum deblending contrast parameter to 0.001. Deblending is a step in SExtractor which splits overlapping objects. When the deblending contrast parameter is set to 0.001, two sources will be considered to be separate only if the integrated pixel intensity of each proposed source

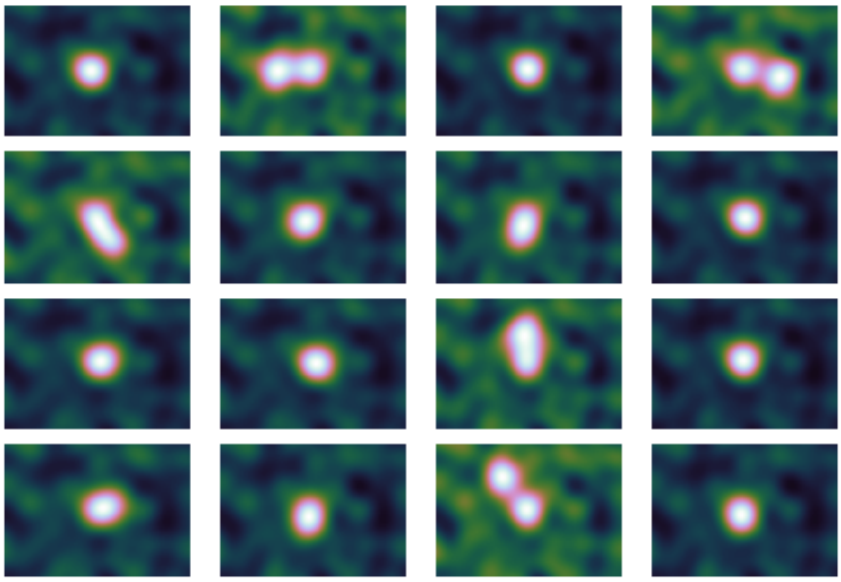


Figure 2.13: 16 dirty images from the 100 dirty images made from simulations for the experiment. These images are zoomed in to the central region, to highlight the location of the two sources. The colour map is CubeHelix (Green 2011).

is greater than 0.001 times the total intensity of them. These parameters were tested so as to best apply SExtractor to the CLEANed images based on our simulated datasets. The positions of point sources found by SExtractor can be used to calculate the source distances, which can then be compared to the distances calculated from the output of BaSC. In BaSC, the source locations are given as the cluster centres according to the clustering output.

The two sets of distances found by SExtractor and BaSC are plotted in Figure 2.14. The two axes represent the simulated and output distances between two point sources. The dotted green line in the figure shows the ideal scenario in which the detected distance is identical to the input. The results of SExtractor are plotted as dashed blue lines. The vertical grey lines represent the sizes of the minor and major axes of the CLEAN restoring beam, which are respectively 0.59 and 0.63 arcsec. We see that SExtractor is unable to distinguish two point sources when their distances are smaller than the size of the restoring beam. Results from BaSC are plotted in solid red lines. All error bars are 90% confidence intervals.

In the comparison, BaSC is able to find two sources in 51 out of 100 cases with a minimum discrimination distance of 0.3 arcsec, whereas SExtractor can only discriminate points separated by no less than 0.6 arcsec away in 27 cases. Since there are 10 datasets for each input distance, the error bar demonstrates the distance range found by these two packages. For a particular distance input, SExtractor has a larger detected distance range, which shows that its accuracy in position detection is inferior to BaSC.

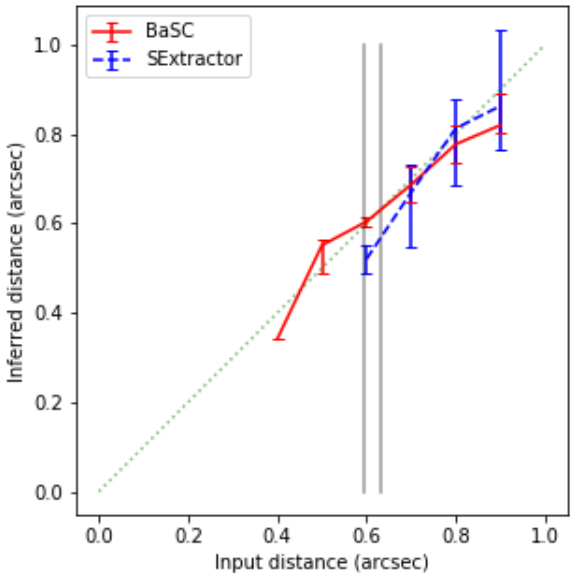


Figure 2.14: Distances found by `SExtractor` and `BaSC` from 100 simulated datasets containing two point sources. The two axes represent the simulated and output distances between two point sources. The dotted green line in the figure shows the ideal scenario in which the output distance is identical to the input. The results of `SExtractor` are plotted in dashed blue lines, with vertical grey lines representing the sizes of the minor and major axes of the CLEAN restoring beam. `SExtractor` is unable to distinguish two point sources when their distances are smaller than the size of the restoring beam. Results from `BaSC` are plotted in solid red lines. All error bars are 90% confidence intervals.

To stretch the capabilities of `BaSC`, we introduced a new dataset. Compared to ALMA, VLA has a dirty beam shape with significant sidelobes if the observation period is short. In the right image of Figure 2.15, the sidelobes of the beam shape are shown around the two central sources. This is due to inadequate uv–coverage of our simulated VLA data. The dirty image demonstrated is noticeably more chaotic than those from ALMA, so we wish to explore whether this disadvantage with dirty images causes `SExtractor` to outperform `BaSC`, since `SExtractor` works on CLEANed images which would not be influenced so much. The dataset used here is simulated in `CASA` using task `simobserve` with a component list based on the distances and angles from the previous dataset[c]. The antenna configuration is changed to VLA A–array for an observation period of 1 hour at 4 GHz. The two points are both 50 mJy and the noise is 2 mJy, with a SNR of 25. The image size of dirty and CLEANed images shown is 300×300,

[c]The actual distances between the two point sources in the VLA simulated data differ slightly from distances in the ALMA data. This is because the position of the off–centre source is calculated by distances and angles in the ALMA data but is then rounded to the nearest pixel, to take advantage of the component list in the task `simobserve`.

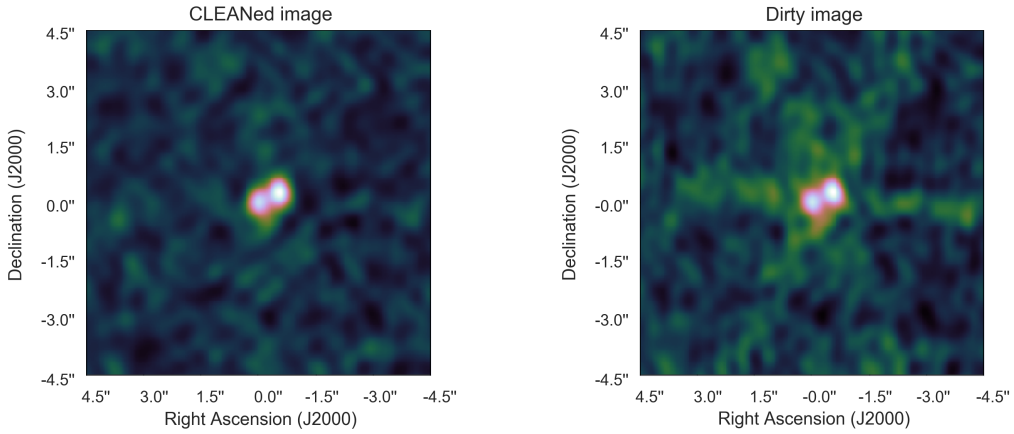


Figure 2.15: Demonstration of the CLEANed image (left) and a dirty image (right) made from VLA A–array simulated data. The sidelobes of the dirty beam shown around the central two sources have been CLEANed in the CLEANed image on the left. `SExtractor` works only on CLEANed images, whereas `BaSC` works on dirty images. The colour map is CubeHelix (Green 2011).

with a pixel size of 0.03 arcsec.

We performed the same numerical experiments on the new VLA dataset. The results are shown in Figure 2.16. The vertical grey lines represent the sizes of the minor and major axes of the CLEAN restoring beam, which are respectively 0.50 and 0.66 arcsec. We find that `BaSC` still outperforms `SExtractor` in estimating the minimum discrimination distance, although both software packages improved their minimum discrimination distance due to an improved SNR. Most distances detected by `SExtractor` are plotted below the green dotted line, or the ideal output. This systematic underestimation of distance using `SExtractor` is due to a spurious 'bridge of fluxes between the two points as parts of the dirty beam are left on the CLEANed image. Consequently `SExtractor` fits Gaussians with closer centres, leading to underestimation of distance. This is ultimately because CLEANed images are not a true representation of the sky. These findings suggest also that deconvolution methods have a significant impact on the scales studied here.

As well as the accuracy of the results, the minimal human intervention required is a further advantage of using `BaSC`. In `SExtractor`, the quality of the CLEAN image has a great impact on source extraction, as our experiment indicates; CLEAN with interactive mode is preferable, which unfortunately requires more human effort. `BaSC` works on dirty images which can be pre–calculated and require no further human intervention. When using `BaSC`, the only human input is an initial estimate of the noise level, which can be approximated by Gaussian fitting. In our newest version of `BaSC`, furthermore, it is not necessary to provide an initial noise value.

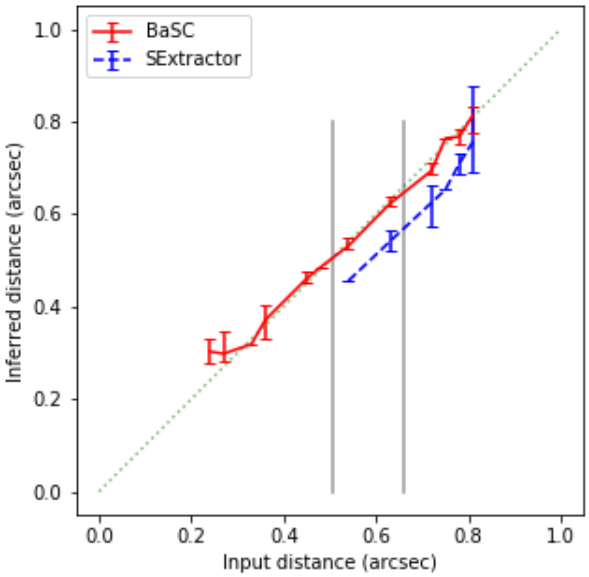


Figure 2.16: Distances found by `SExtractor` and `BaSC` using VLA simulated data. `SExtractor` is unable to distinguish two point sources when their distances are smaller than the size of the restoring beam. The results are plotted in dashed blue lines. Results from `BaSC` are the solid red lines. All error bars are 90% confidence intervals.

2.5.3 Comparison between `BaSC` and `Aegean`

The dataset used in this experiment to compare the performance of `BaSC` and `Aegean`[d] is the same ALMA data used for the comparison between `BaSC` and `SExtractor`.

For the noise level input, `Aegean` can either take a fixed noise level or use an internal command `BANE` to derive its noise model. After testing both methods, we found that the former is more accurate in source separation. Although the largest distance between two point sources in our dataset is 0.9 arcsec, `Aegean` still finds outliers with a distance greater than 1 arcsec as sources. Hence, outliers were removed by manual intervention.

`Aegean`'s performance depends also on the CLEANed image (Hancock et al. 2018), and Figure 2.17 reveals that `Aegean` cannot distinguish two point sources when their distances are smaller than the size of the CLEAN restoring beam. The distance accuracy does not improve very much as the distance increases, however, giving worse performance than `SExtractor`.

[d] Available at https://github.com/PaulHancock/Aegean

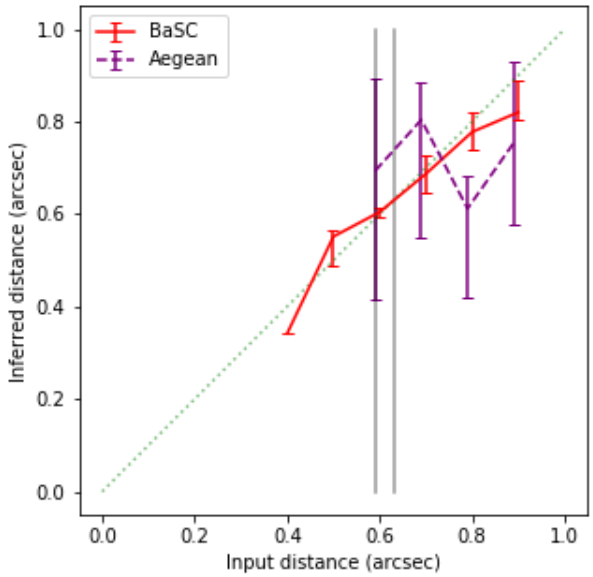


Figure 2.17: Distances found by Aegean and BaSC using the same ALMA simulated data. The results of Aegean are plotted in dashed purple lines; vertical grey lines represent the sizes of the minor and major axises of the CLEAN restoring beam. Aegean is unable to distinguish two point sources when their distances are smaller than the size of the CLEAN restoring beam. Results from BaSC are plotted in solid red lines. All error bars are 90% confidence intervals.

2.5.4 BaSC performance

We next designed an experiment to see whether the performance of BaSC improves or deteriorates if we change the flux contrast between the two point sources. Based on the 100 ALMA simulated data, we fix the flux of the off–centre point but increase the flux of the central points by factors of 10, 20, 30 and 40 respectively. The distances and angles between them remain unchanged. We apply BaSC to these modified simulated data.

Figure 2.18 shows the performance of BaSC with datasets having different source flux contrasts. The dashed red lines in the four plots represent the same BaSC results on the previous simulated ALMA datasets with equal source fluxes; this is identical to the solid red line in Figure 2.14 and 2.17. The solid red line in each plot represents the results of BaSC with the new simulated ALMA datasets having unequal source fluxes. The vertical solid lines represent the sizes of the minor and major axes of the CLEAN restoring beam, which are 0.59 and 0.63 arcsec respectively. As the flux contrast increases from 10 to 40 times, the minimum discrimination distance decreases.

When the flux of the brighter source increases, its position is easier to detect due to its high flux, so that the flux uncertainty is the only significant source of error. Extraction of the

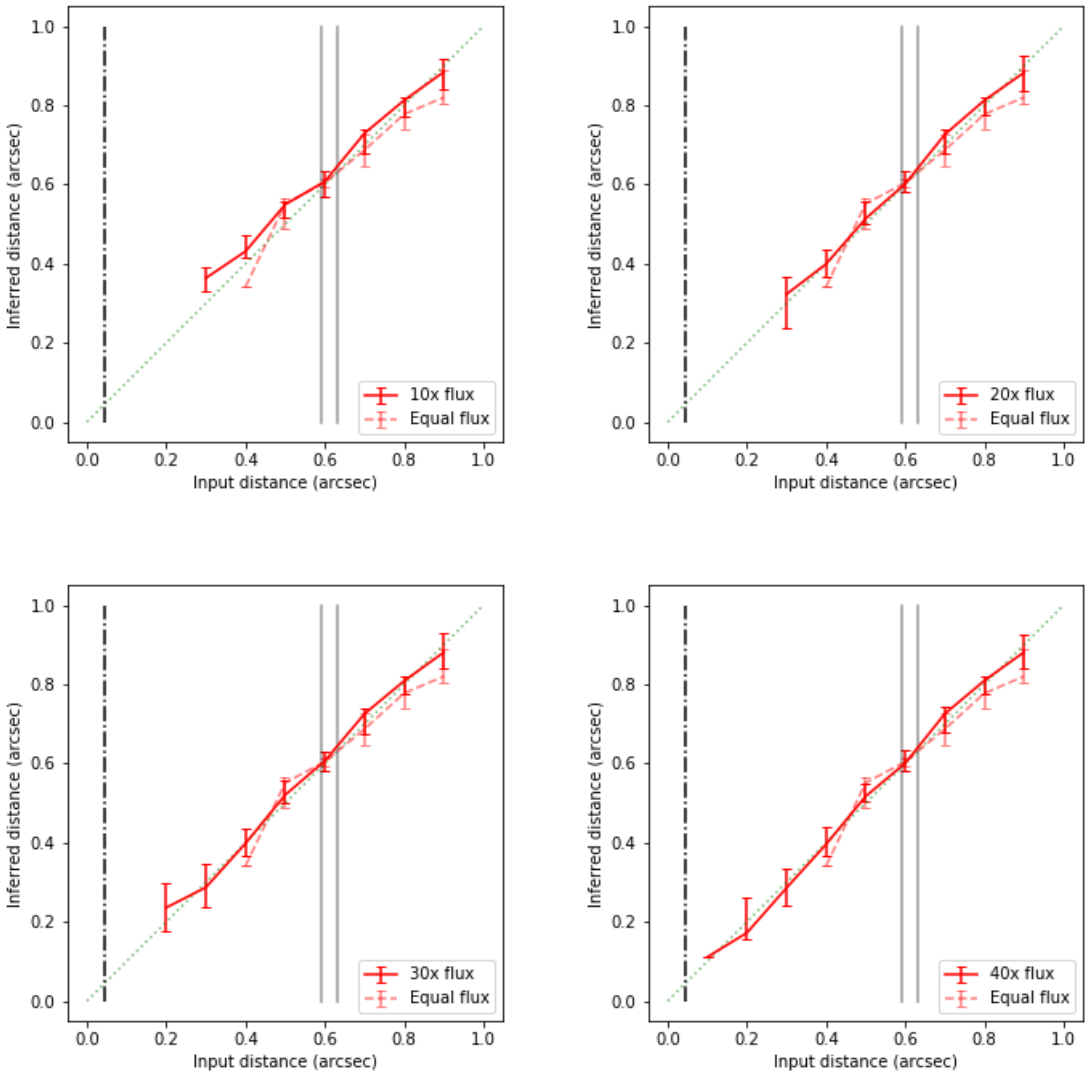


Figure 2.18: BaSC performance on datasets with different source flux contrasts. The dashed red lines in the four plots represent the identical BaSC results on the previous simulated ALMA datasets with equal source fluxes. The solid red line in each plot represents the BaSC results on the new simulated ALMA datasets with unequal source fluxes. The vertical dot–dashed line shows the limit of position accuracy of the dimmer source, or the separation limit of the two sources. As the flux contrast increases from 10 to 40 times, the minimum discrimination distance decreases. As the the flux contrast increases to 40, the minimum discrimination distance is very close to the theoretical separation limit shown by the vertical dot–dashed line. All error bars are 90% confidence intervals.

dimmer source can therefore be regarded as an independent single point source problem with a noise level equal to 2σ. According to Table 2.1, the dimmer source with an SNR of 10 has a theoretical Bayesian resolving power of 2.351 cells, or 2.351×10^{-2} arcsec. Twice this value, or 4.702×10^{-2} arcsec, is marked by the vertical black dot–dashed line in each plot in Figure 2.18, showing the limit of position accuracy of the dimmer source, or the separation limit of the two sources. In Figure 2.18, as the flux contrast increases to 40, the minimum discrimination distance falls and approaches the theoretical separation limit marked by the vertical dot–dashed line.

2.6 Conclusion

In this chapter we proposed a new Bayesian source extraction method, and introduced the software BaSC which implements it. We began with Bayes' theorem, followed by theoretical derivations underlying the method of source extraction. By introducing MCMC and using the chosen MCMC driver BayeSys, we implemented the method on electronic computers and and obtained output models. The output data from BayeSys then has to be clustered, and DBSCAN and HDBSCAN proved to be suitable clustering algorithms in this application so as to provide the desired source extraction information. The software BaSC was designed to implement all of these stages.

We performed experiments to compare the performance of BaSC with two other source extraction packages, SExtractor and Aegean. We found that BaSC is superior to SExtractor and Aegean at the specific task of discriminating between two nearby point sources. BaSC works on intermediate image products from the visibility data, whereas the other two packages work on reconstructed images only, such as CLEANed images. The minimum discrimination limit of SExtractor and Aegean is limited by the size of the CLEAN restoring beam, but BaSC can distinguish sources with a separation smaller than that.

We derived the theoretical resolving power of our Bayesian method, and tested it with single point–source simulated data. In situations with two point sources in which one source is much brighter than the other, BaSC's ability to discriminate two points improves with an increase in the flux contrast, and its minimum discrimination distance approaches the theoretical resolving power. With the easier position detection and better flux subtraction of the brighter source, the extraction of the dimmer source can then be considered as a decoupled problem, similar to the single point–source model. These numerical experiments found that the minimum discrimination distance of two point sources with large flux contrast is limited only by the theoretical Bayesian resolving power.

A further advantage of our method is the minimal human intervention. In the most up-to-

date version of BaSC, no human input is needed except to input the image. In contrast, unlike other packages working on CLEAN images, BaSC works with intermediate image products: dirty image, dirty beam and primary beam. BaSC thereby circumvents computational and human effort to perform the deconvolution.

There are still limitations in BaSC. Its performance deteriorates when the source number exceeds 30. With a low SNR, BayeSys output is too complex to facilitate clustering by the chosen clustering algorithms. The present method is also restricted to images with small field view, and should be reconsidered when dealing with large fields, where the dirty beam is position-dependent rather than fixed. To overcome these problems calls for further research.

In the future, we propose to generalise our Bayesian method so as to detect sources having arbitrary shapes. We may assume that all sources can be represented by a limited number of two-dimensional Gaussians, having major and minor axes as parameters. For the clustering algorithm, a machine learning approach could prove fruitful.

Chapter 3

Optimal Gridding and Degridding

The Fourier transform relation between the visibility data and the sky brightness explained in Section 1.2 is fundamental to imaging procedures in radio interferometry. The number of visibilities N_v increases along with the increasing number of baselines, frequency channels and demand of observation time. The computationally expensive DFT operation hence has been substituted by FFT, which reduces the computational complexity of making an image of N_l pixels by N_m pixels from $O(N_v N_l N_m)$ to $O\left(N_l N_m \log(N_l N_m)\right) + O(N_v)$ (James W. Cooley 1965; Heideman et al. 1985; Smith et al. 2017).

The visibilities are not sampled on a Cartesian grid, which the FFT algorithm would require. This problem arises also, for example, magnetic resonance imaging (MRI) and X–ray diffraction computerised tomography (CT), as the data are again not sampled on a Cartesian grid but FFT is required to reconstruct the target image (Schomberg & Timmer 1995). The obvious solution is to interpolate the visibility data onto a Cartesian grid and then apply the FFT.

Nearest–neighbour method was an early application of gridding in radio astronomy, and an example of its implementation is given by Hogg et al. (1969). In this thesis, nearest–neighbour interpolation was used to generate Figure 1.5, in which many point–like artefacts can be found around the outer edge of the discrepancy between the DFT and FFT images. More elaborate interpolation methods include 'cell summing' in Mathur (1969), and 'radial interpolation', proposed by Thompson & Bracewell (1974). These interpolation methods are no longer used because they proved poor at suppressing aliasing. Aliasing is an unwanted phenomenon in which brightness, including noise outside of the field of interest, appears within (Greisen 1979; Briggs et al. 1999). It is discussed in Section 3.1.

The most common method for gridding is to convolve the visibility data with a chosen gridding function, as stated in Section 1.3. As early as 1961 Elizabeth Waldram, in the Radio Astronomy Group at the Cavendish Laboratory, Cambridge, was using gridding functions that included the Gaussian, the sinc function, and Gaussian times sinc function, according to her work notes, although this work was unpublished. Brouw (1971) later used a Gaussian gridding function with the east–west synthesis array at Westerbork. Many gridding functions have since been explored. The gridding function is sometimes referred as the gridding convolution function or gridding convolution kernel, or simply the kernel.

To suppress the effects of aliasing in the image plane, different choices of gridding function have been considered in (Greisen 1979) and (Schwab 1978). By optimising a quantity proposed by Brouw (1975) as a measure of the suppression of aliasing, the prolate spheroidal wave function of order 0 (Slepian & Pollak 1961; Landau & Pollak 1961) proved to be the optimal gridding function (Brouw 1975; Schwab 1984). Schwab (1980, 1984) then argued for a modified optimality criterion which singled out and the spheroidal function (Stratton 1935) as the best gridding function. The spheroidal function is implemented in `AIPS` and `CASA`. The spheroidal function is not easily calculated, and a pre–computed look–up table is usually provided (Thompson et al. 2017; Jackson et al. 1991).

Use of the gridding function is not confined to the making of dirty images and dirty beams; it is also used in the degridding process explained in Section 1.4. Degridding is essential to deconvolution methods such as Cotton–Schwab CLEAN (Schwab & Cotton 1983) and the Maximum Entropy method (Gull & Daniell 1978). In the major cycle of Cotton–Schwab CLEAN, for instance, after applying an FFT to the image model, the resulting visibility model is on Cartesian grids, and needs to be correlated with the gridding function in order to assign it back to where the visibility data were originally sampled. In summary, the gridding function is required in both gridding and degridding in imaging procedures, and its choice consequently influences the quality of the image products obtained and all analyses based on them.

Since degridding process is usually adopted during the self–calibration procedure (Cornwell & Fomalont 1999), the choice of gridding function also will influence the accuracy of the self–calibrated visibility data, and consequently affect the accuracy of the final image products and relevant data analysis.

Given the major role of the gridding function in the imaging and self–calibration processes, a more rigorous optimality criterion for the gridding function is called for. The gridding function was originally introduced to further the replacement of DFT by FFT, so any new gridding function should minimise the difference between the DFT and FFT dirty images. By optimising and implementing such a gridding function, we will obtain more accurate image products compared to corresponding DFT results. Also, image–based data analysis, such as the

source extraction explained in Section 2, should also benefit from improved image accuracy.

This chapter develops the subject of gridding functions, and also the processes of gridding and degridding. Section 3.1 introduces the gridding process and then discusses aliasing effects due to the gridding and inverse FFT process, followed by a brief introduction to spheroidal functions. In Section 3.2 a new gridding function, the 'least–misfit gridding function' is proposed, based on minimising the difference between the DFT and FFT dirty images. The theory and its computational implementation are explained in full detail. In Section 3.3 the spheroidal function is reevaluated according to the criterion of minimising the difference between the DFT and FFT dirty images. The performance of the spheroidal function and least–misfit function will be compared both practically and theoretically, using the optimisation criteria, in Section 3.4. Section 3.6 demonstrates the degridding performance using the least–misfit gridding function. Practical implementation of the least–misfit functions in imaging is shown in Section 3.5, with calculation of the corresponding computational cost. Conclusions are presented in Section 3.7.

3.1 The gridding process and aliasing effects

The gridding process can be viewed mathematically as a combination of a convolution and a sampling process (O'Sullivan 1985); the convolution between the gridding function and the visibility data results in a continuous function, which need be sampled only at a uniformly distributed grid. In practice, the convolution is conducted only at each grid point (Thompson et al. 2017). One–dimensional notation $C(u)$ will be used for simplicity, since the gridding function is separable.

The sampling process can be written as a multiplication between the convolved visibilities and a so–called *Shah* function $\text{Ш}(u)$. The Shah function is a sum of equally spaced delta functions:

$$\frac{1}{\Delta u}\text{Ш}\left(\frac{u}{\Delta u}\right) \equiv \sum_{k=-\infty}^{\infty} \delta(u - k\Delta u) \tag{3.1}$$

where Δu is the grid size. By rewriting the delta functions in their spectral representation, it can be shown that the inverse Fourier transform of this is:

$$\text{Ш}(l\Delta u) = \frac{1}{\Delta u}\sum_{k=-\infty}^{\infty} \delta\left(l - \frac{k}{\Delta u}\right) \tag{3.2}$$

which is also a Shah function, but with a different sampling frequency.

To treat the gridded visibilities, an inverse Fourier transform is applied on the image plane as:

$$\mathscr{F}^{-1}\left[(C(u) * V(u))\frac{1}{\Delta u}\text{Ш}(\frac{u}{\Delta u})\right] = \{\mathscr{F}^{-1}[C(u)]I_D(u)\} * \text{Ш}(l\Delta u) \tag{3.3}$$

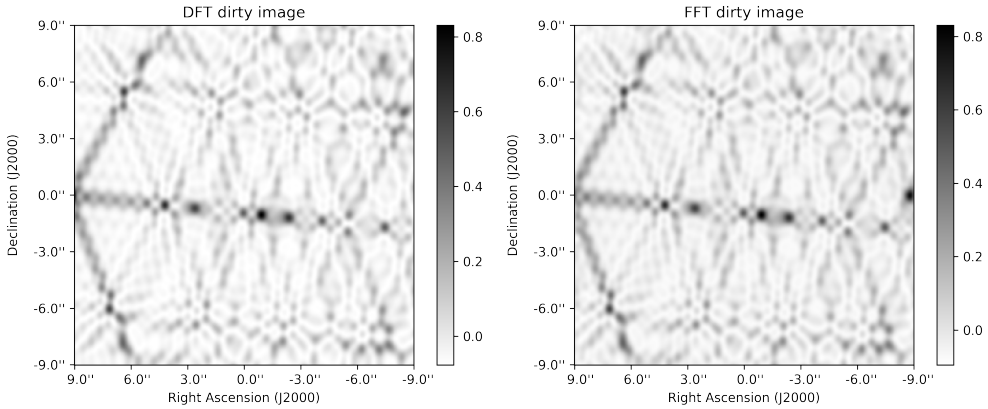


Figure 3.1: Demonstration of the aliasing effect due to the periodicity of FFT. DFT dirty image (top left) and FFT dirty image (top right) on the simulated point–source visibility data. The alias is obvious in the FFT dirty image, with a brightness of 0.82 Jy.

where $\mathscr{F}^{-1}(\cdot)$ represents the inverse Fourier transform.

Greisen (1976) proposed that the FFT image results be divided by the inverse Fourier transform of the chosen gridding function $C(u)$, so as to cancel the effects brought by $C(u)$ and obtain the correct fluxes. The reciprocal of the inverse Fourier transform of the gridding function $C(u)$ is usually referred as the 'correcting function', which is to be multiplied by the FFT image result (Briggs et al. 1999).

Since $V(u)$ is equivalent to the sampled and weighted visibilities denoted by V^{SW} in Chapter 1, its inverse direct Fourier transform is the DFT dirty image $I_D(u)$. By applying FFT to replace DFT on the gridded visibilities in Equation 3.3, and then correcting the results of FFT with the corresponding correcting function, we obtain the FFT dirty image $\widehat{I_D}(u)$, which is an approximation to $I_D(u)$.

Since the Shah function $\text{Ш}(l\Delta u)$ is periodic, the FFT dirty image $\widehat{I_D}(u)$ is also periodic, with a period $1/\Delta u$. If the finite support of the dirty image is larger than $1/\Delta u$, 'aliasing' occurs. It follows that Δu should be chosen carefully so as to avoid aliasing.

Quite apart from the introduction of the shah function, the periodic nature of FFT itself gives rise to aliasing. As an example, a DFT dirty image of a simulated VLA A–array data is shown in the left map in Figure 3.1. There is only one point source, located at $(9.3'', 0'')$ with a flux of 2 Jy. Since the pixel size is 0.03 arcsec and the image size is 600×600, the source is 310 pixels distant from the image centre, therefore 10 pixels distant from the left image edge. No noise was added to this simulated data. The periodicity of FFT brings the point source to $(-8.7'', 0'')$ with a brightness of 0.82 Jy. The gridding function used is the least–misfit gridding function with a width of 7, which will be introduced and discussed in full in Section 3.2.

Aliasing is only one of many causes of discrepancy between DFT and FFT dirty images.

O'Sullivan (1985) found that use of the gridding function gives smaller errors around the image centre than near to the edge. As a result a larger dirty image is always made, followed by an image cropping process to deal with the larger errors at the image edge. O'Sullivan (1985) and Jackson et al. (1991) proposed that an FFT image should be made that is twice as large as the intended image, and then the outer half of it should be discarded.

The gridding function is also required to have finite support, so that the computational cost of the convolution is not unduly large. (O'Sullivan 1985) proposed the infinite sinc function as the gridding function, but its performance clearly degrades when it is truncated. Schwab (1978); Greisen (1979) considered suppressing the effects of aliasing by choosing an appropriate gridding function, and set out some choices.

3.1.1 The aliasing suppression ratio

We first explain the normalised image plane coordinate used in this chapter. In Figure 3.2, the upper coordinate system is formed by the direction cosines (l, m) explained in Chapter 1. Here $l_{\min}$, $m_{\min}$, $l_{\max}$ and $m_{\max}$ define the size of the two–dimensional field of interest, centred at the phase centre $(0, 0)$. The lower normalised coordinate system shows the corresponding normalised field size defined by $x \in [-x_0, x_0]$ and $y \in [-y_0, y_0]$. The dashed square shows the normalised size of the dirty image, to be restricted to the range $[-0.5, 0.5]$ on both axes. The values x_0 and y_0 represent the discarding points on both axes. x_0 and y_0 are chosen between 0 and 0.5. If x_0 and y_0 are both set to 0.5, no image cropping is performed. If x_0 and y_0 are set to 0.25, a double–sized dirty image will be made, and its outer half will then be discarded so as to retain only the inner half of the image bounded by $[-x_0, x_0]$ and $[-y_0, y_0]$. We usually take x_0 and y_0 as the same, but this does not mean that the angular size of both sides of the field are identical. In this chapter, we use 'normalised map coordinate' and 'normalised image plane coordinate' interchangeably.

Next, we revisit the aliasing suppression ratio, which measures the aliasing–suppression capability of a gridding function $C(u)$ on the image plane. If a source is lying outside the field of interest at position x $(x > 0.5)$, it can alias into the field at position $(x + 0.5) \mod 1 - 0.5$, and vice versa. The aliasing suppression ratio $q(x)$ is defined as the intensity of an alias at x $(x > 0.5)$ from a source located within the field at $[((x + 0.5) \bmod 1) - 0.5]$, divided by the source's intensity (Greisen 1979). It can be written as

$$q(x) = \frac{c(x)}{c[((x + 0.5) \bmod 1) - 0.5]} \tag{3.4}$$

where $c(x)$ is the real part of the inverse Fourier transform of $C(u)$, with $x \in [-0.5, 0.5]$ representing all positions on the dirty image to be made in one dimension. Here, $x = 0$ and

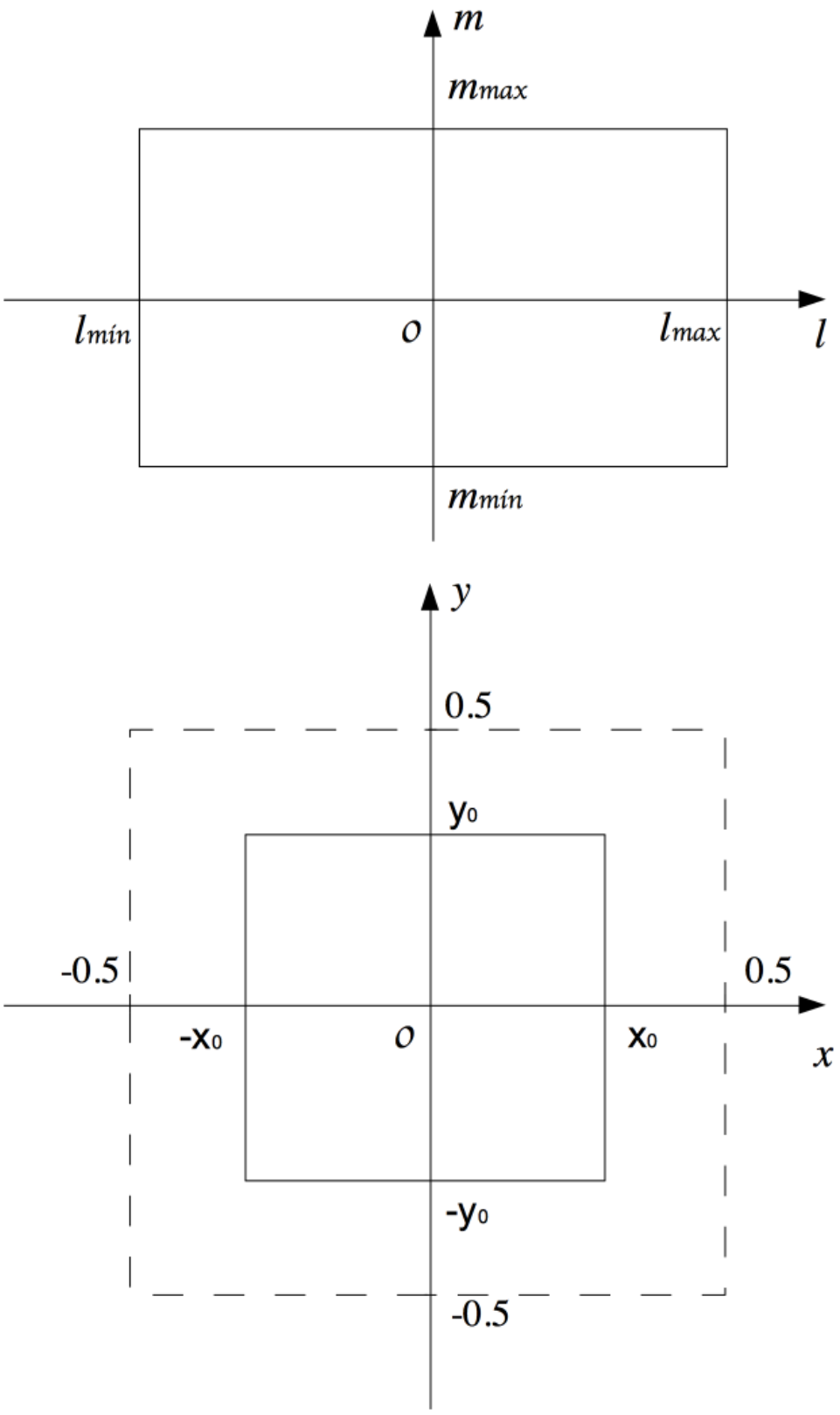


Figure 3.2: The coordinate system formed by the direction cosines (l, m) (top) and the normalised coordinate system (x, y) (bottom). The upper coordinate system is centred at the phase centre $(0, 0)$. $l_{\min}$, $m_{\min}$, $l_{\max}$ and $m_{\max}$ define the size of the two–dimensional field of interest. The lower, normalised, coordinate system shows the corresponding normalised field defined by $x \in [-x_0, x_0]$ and $y \in [-y_0, y_0]$.

$x = \pm 0.5$ correspond to the centre and both edges of the dirty image. The position $x = 1$ represents the location one image size distant from the image centre $x = 0$.

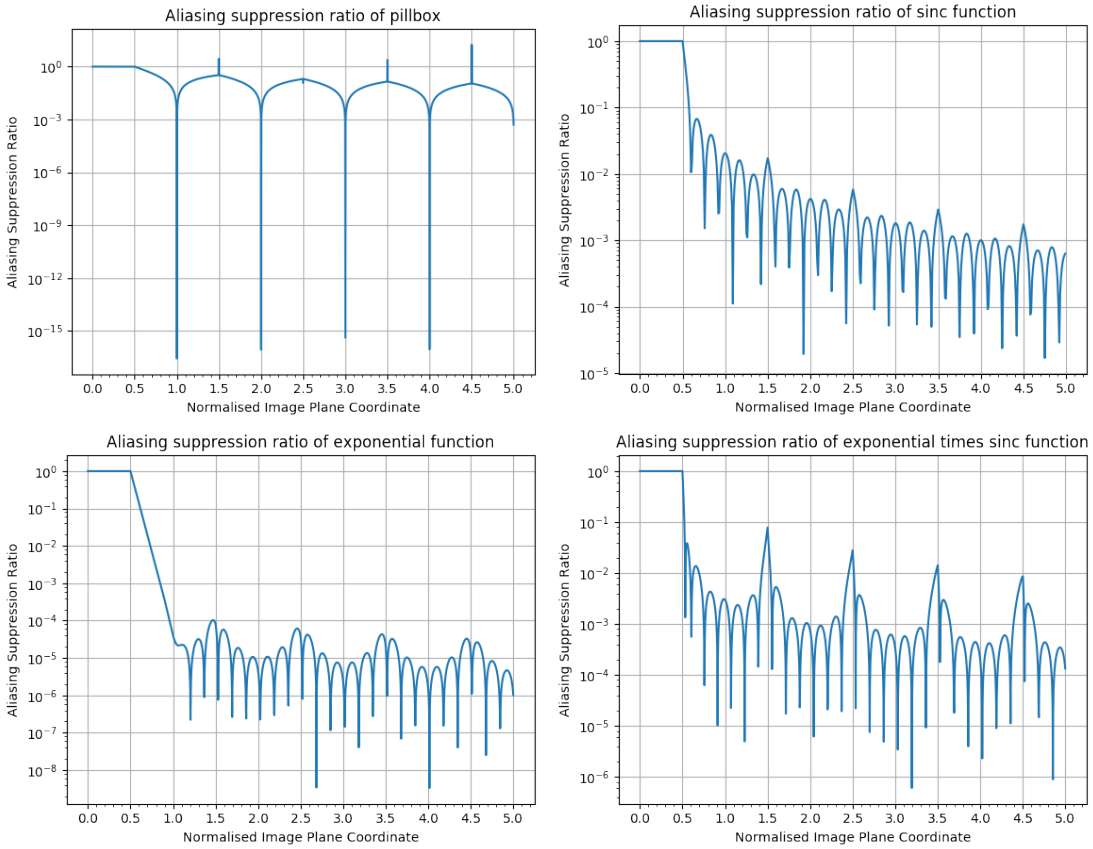


Figure 3.3: Aliasing suppression ratio for four common gridding functions: (1) pill box: $C_1(u) = 1$; (2) sinc function: $C_2(u) = \mathrm{sinc}(u)$; (3) exponential function: $C_3(u) = \exp(-|u|^2)$; (4) sinc times exponential function: $C_4(u) = \exp - \left(\frac{|u|}{2.52}\right)^2 \mathrm{sinc}\left(\frac{u}{1.55}\right)$. All functions are confined to a window width of $W = 6$. The x–axis represents the normalised image plane coordinate, and the y–axis shows values of the aliasing suppression ratio.

We reproduce the aliasing suppression ratio plots of the following four common choices of gridding function in Figure 3.3:

- pill box: $C_1(u) = 1$;
- truncated sinc function: $C_2(u) = \mathrm{sinc}(u)$;
- truncated exponential function: $C_3(u) = \exp(-|u|^2)$;
- truncated sinc times exponential function: $C_4(u) = \exp - \left(\frac{|u|}{2.52}\right)^2 \mathrm{sinc}\left(\frac{u}{1.55}\right)$.

The constants are chosen according to the recommendations in Briggs et al. (1999). The x–axis represents the normalised image plane coordinate, and the y–axis shows the aliasing suppression ratio. All functions are confined to a window width of $W = 6$. As a result of the periodic nature of FFT, with a period of 1, sources close to integer numbers, e.g. $x = 1$, will

be aliased into positions around the image centre $x = 0$; sources close to half integer numbers, e.g. $x = 1.5$, will be aliased to the image edge $x = 0.5$. In the latter case, cropping of a portion of the final FFT dirty image from the edge can remove the aliasing effects. For the former case, however, a good gridding function is needed: the aliasing suppression ratio close to integers should be strong enough to mitigate the intensities of the aliases.

Upon looking at Figure 3.3, we observe two important aspects of the suppression ratio plots:

- The values of the aliasing suppression ratio: smaller ratio gives better aliasing suppression performance on the image plane, especially around integer numbers on the normalised image plane coordinate.
- The fluctuation of the aliasing suppression ratio: less volatile ratios gives more stable suppression performance.

The pillbox was a convenient choice of gridding function because of its simplicity, but it gives rise to serious aliasing effects in the image. A source located as far as 5 image sizes from the image centre ($x = 5$) will alias into the image centre with approximately 10^{-3} of its original flux strength. If the source is much stronger than the sources within the field, this will then stand out in the image as a 'ghost source'. The other three gridding functions suppress aliasing significantly better than the pillbox.

The truncated exponential function gives relatively strong and stable aliasing suppression; its suppression ratio is around 10^{-5} to 10^{-7} with small fluctuations. The truncated sinc function's suppression ratio decreases smoothly as the distance from the image centre increases, but the relatively large values, especially closer to the image centre, make aliasing a concern. The suppression ratio of truncated sinc multiplied by the exponential function has peaks around half–integers, but is small around integer numbers. The image parts around half–integers will be discarded, and the suppression ratio values around integer numbers are around 10^{-3} to 10^{-5}, which is worse than the truncated exponential function.

These four gridding functions were all proposed by heuristic reasoning followed by trial and error. A systematic method is needed that provides a gridding function by optimising its performance.

3.1.2 Spheroidal functions

If A represents the field of interest, the optimality criterion proposed by Brouw (1975) is to find the gridding function $C(u)$ which maximises the quantity

$$R = \frac{\int_A |c(x)|^2 \mathrm{d}x}{\int_{-\infty}^{\infty} |c(x)|^2 \mathrm{d}x} \tag{3.5}$$

This criterion gives a gridding function containing the least energy outside the region A (Jackson et al. 1991). It leads to the zero–order prolate spheroidal wave function.

Schwab (1980, 1984) modified the quantity to be optimised with an additional weight function $w(x) = |1 - (2x)^2|^\alpha$ as follows:

$$R_\alpha = \frac{\int_A |c(x)|^2 w(x) \mathrm{d}x}{\int_{-\infty}^{\infty} |c(x)|^2 w(x) \mathrm{d}x} \tag{3.6}$$

For $\alpha = 0$, $w(x) = 1$ so that the optimised gridding function is still the zero–order prolate spheroidal wave function. For general values of α, Schwab (1980, 1984) found that the gridding function which extremises this expression is a more general form involving spheroidal functions:

$$C(u) = |1 - \eta^2(u)|^\alpha \psi_{\alpha 0}(\pi W/2, \eta(u)), \qquad |u| < \frac{W}{2} \tag{3.7}$$

where $\psi_{\alpha 0}(\pi W/2, \eta(u))$ is a zero–order spheroidal function and $\eta(u) = 2u/W$ (Schwab 1984). Its numerical approximation with different values of W and α was given by Schwab (1981) and has been used ever since. The weight function was introduced for better suppression of aliasing closer to the image centre. Cropping might still be needed nearer the edge of the image.

Typical values of α and W are $\alpha = 1$ and $W = 6$ (Briggs et al. 1999). These are implemented in `AIPS` as the default gridding function. The Kaiser–Bessel window function is considered a good approximation to the spheroidal function with lower computational cost (Jackson et al. 1991).

The gridding process with an odd value of the width is not as straightforward as with even support; see Figure 1.6. In Figure 3.4 we demonstrate how a gridding function $C(u)$ works with $W = 5$. For an off–grid visibility $V(u_0)$ lying closer to $([u_0] - u_0)$ than to $([u_0] - u_0 + 1)$, its value will be assigned to three grid points on the left and two grid points on the right with weights $\{C([u_0] - u_0 + r)\}$, where $r = -2, -1, 0, 1, 2$. Here $[u]$ represents the largest integer less than u. Otherwise, if u_0 is closer to $C([u_0] - u_0 + 1)$, there are three grid points on the right and two grid points on the left. The corresponding weights change to $\{C([u_0] - u_0 + r)\}$, where $r = -1, 0, 1, 2, 3$.

The spheroidal function was proposed so as to reduce aliasing, particularly around the central area of the image, so it should yield a good aliasing suppression ratio. Figure 3.5 shows the aliasing suppression ratio of the zero–order spheroidal functions with $\alpha = 0, 1$ and different values of W. In our numerical experiments, instead of using rational approximations to spheroidal functions, we use the '`pro_ang1`' function from the `Python` package `scipy` for greater accuracy. The root mean square (RMS) difference between this function and the numerical approximation proposed by Schwab (1981) is less than 10^{-6}.

A periodic pattern is visible in the suppression ratios in both plots; the suppression ratio falls close to integer values and increases towards half integers, and is symmetric about the integers. This is because sources that alias into positions around the image centre are suppressed more

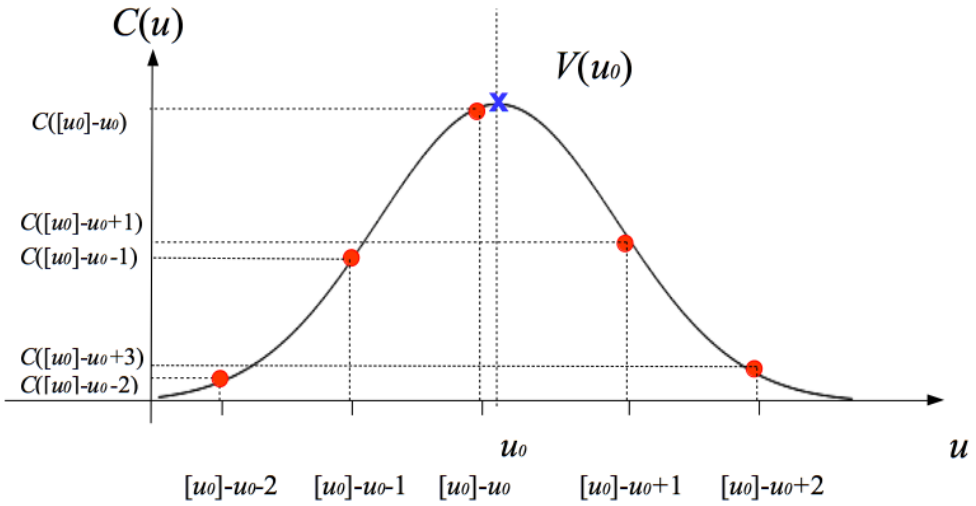


Figure 3.4: Demonstration of the gridding process with an odd value of the width $W = 5$. For an off–grid visibility $V(u_0)$ lying closer to $([u_0]-u_0)$ than to $([u_0]-u_0+1)$, its value is assigned to three grid points on the left and two grid points on the right with weights $\{C([u_0]-u_0+r)\}$, where $r = -2,-1,0,1,2$; $[u]$ represents the largest integer less than u.

strongly than those alias into positions around the image edge. Image cropping is therefore necessary when using the spheroidal function as a gridding function, to remove the aliasing effects around the edges of the image. As an example, for the spheroidal function with $\alpha = 1$ and $W = 10$, a source at $x = 2.0$ will be aliased into the image centre with 10^{-9} of its original brightness, small enough to be neglected. If there are sources close to $x = 1.5$, these will alias to the edge of the image with a suppression ratio of 10^{-3}. If these sources are 10^3 times brighter than those within the field of interest, they will become ghost sources in the FFT dirty image. If $x \in (0.4, 0.5]$ is thrown away so as to discard aliasing at the image edge, the cropped image $x \in (0, 0.4]$ should be the same size as our field of interest.

The suppression ratios for both spheroidal functions decrease as W increases, suppressing the aliasing better. There is also a decreasing trend as the distance from the image centre increases, which indicates that these functions suppress aliasing progressively better as x increases. The function with $\alpha = 1$ has a greater decreasing slope than for $\alpha = 0$ at each value of W. The spheroidal function with $\alpha = 1$ therefore outperforms that with $\alpha = 0$, and it will be the benchmark for comparison in this chapter.

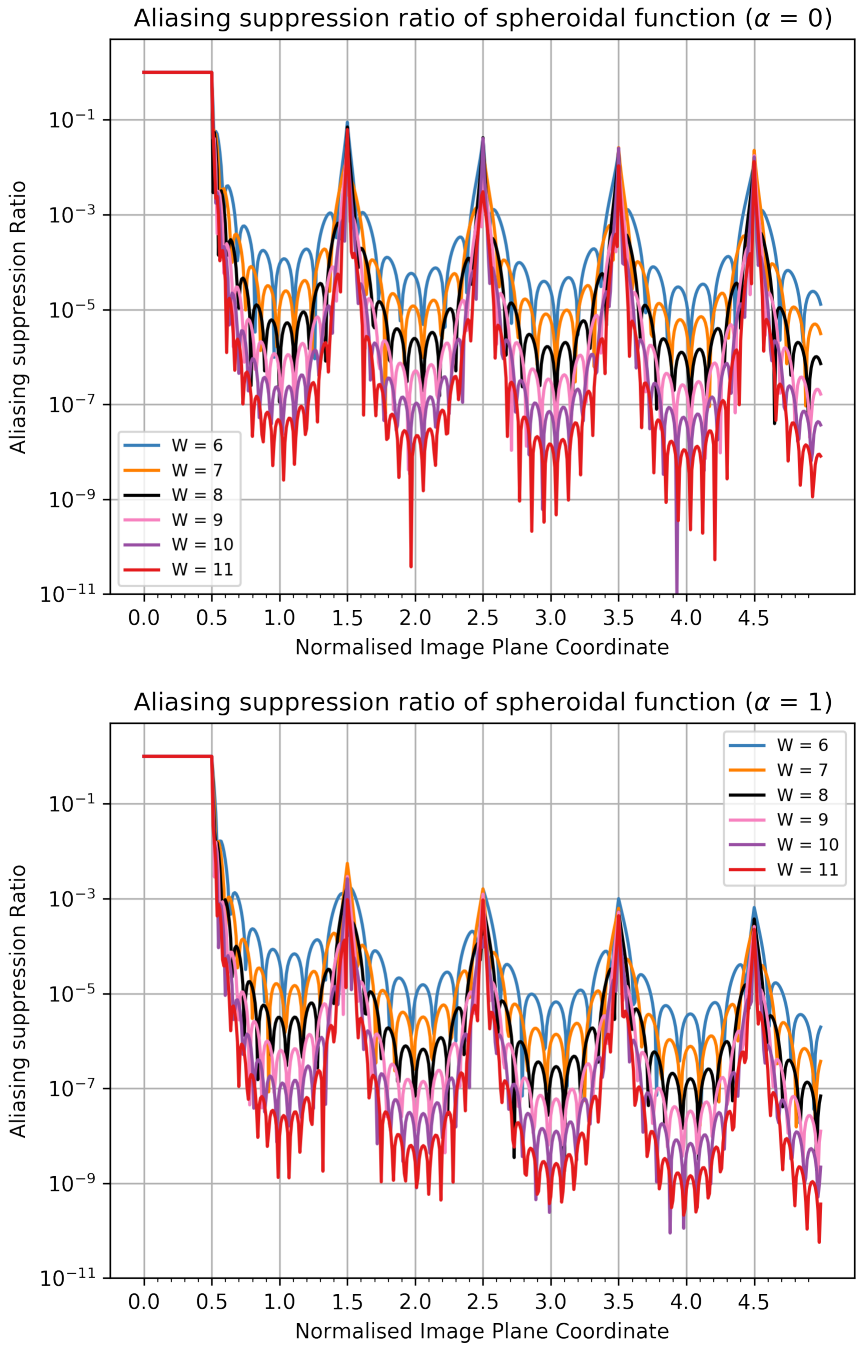


Figure 3.5: The aliasing suppression ratio for the spheroidal functions with $\alpha = 0, 1$ and differing values of W.

3.2 Least–misfit gridding function

Given infinite computing power, DFT would be preferable to FFT, since no information would be lost due to gridding or FFT. We therefore propose a criterion which minimises the difference between the DFT and FFT dirty images. This difference is also referred to as the 'image misfit' or 'map misfit'.

The DFT dirty image defined in Equation 1.11 can be written in the present one–dimensional notation as:

$$I_D(x) = \sum_{k=1}^{M} w_k V_k \exp(i2\pi u_k x) \tag{3.8}$$

where $w_k = 1/\sigma_k^2 > 0$ is a natural weight defined by the standard deviation σ_k of the noise in the visibility data V_k, and M denotes the number of visibilities.

To obtain the corresponding FFT dirty image $\widehat{I_D}(x)$ with size N_x, the visibilities must be gridded to obtain the gridded data G_n, after which the FFT is applied:

$$\widehat{I_D}(x) = h(x) \sum_{n=-N_x/2}^{N_x/2-1} G_n \exp(i2\pi n x) \tag{3.9}$$

where $h(x)$ is the correcting function. The coefficient G_n is written as

$$G_n = \sum_{k=1}^{M} w_k V_k C(n - u_k) \tag{3.10}$$

The gridding function $C(u)$ is real and symmetric with a support of W unit–width cells. If W is even, each visibility V_k is assigned with differing weights obtained from $C(u)$ onto W consecutive grid points $n \in S_k$, where

$$S_k = \{[u_k] + r\}, r = -W/2 + 1, ..., W/2 \tag{3.11}$$

The integer W is chosen by users. When W is odd and $W \geq 3$ with $u_k - [u_k] > 0.5$, r changes in S_k to $r = (3-W)/2, ..., (W+1)/2$; if $u_k - [u_k] \leq 0.5$ then r changes to $r = (1-W)/2, ..., (W-1)/2$.

Our task, for a given W, is to determine the combination of gridding and correcting functions which minimises the difference between the FFT dirty image $\widehat{I_D}$ and the DFT dirty image I_D.

3.2.1 Theory

The idea of minimising the misfit between the DFT and FFT dirty image was first proposed and implemented in Tan (1986). This work will be extended with new results below, including a theory of the gridding function. We begin by writing the error resulting from the use of $\widehat{I_D}$

instead of I_D as

$$e(x) \equiv I_D(x) - \widehat{I}_D(x) = \sum_{k=1}^{M} w_k \underbrace{V_k \exp(i2\pi u_k x)}_{a_k^*} \underbrace{\left\{1 - h(x) \sum_{n \in S_k} C(n-u_k) \exp[i2\pi(n-u_k)x]\right\}}_{b_k} \tag{3.12}$$

From the Cauchy–Schwarz inequality $|\mathbf{a}\cdot\mathbf{b}|^2 \le |\mathbf{a}\cdot\mathbf{a}||\mathbf{b}\cdot\mathbf{b}|$, with $\mathbf{a}\cdot\mathbf{b} \equiv \sum w_k a_k^* b_k$, it follows that:

$$e^2(x) \le \left(\sum_{k=1}^{M} w_k |V_k|^2\right) \underbrace{\left(\sum_{k=1}^{M} w_k \left|1 - h(x) \sum_{n \in S_k} C(n-u_k) \exp[i2\pi(n-u_k)x]\right|^2\right)}_{\ell(x)} \tag{3.13}$$

Since the term $\sum w_k |V_k|^2$ depends only on the data and the weights, our task is reduced to keeping the other factor $\ell(x)$ small.

By writing the fractional offset part of u as $\nu = u - [u] \in [0, 1]$, we have

$$\sum_{n \in S_k} C(n-u_k) \exp[i2\pi(n-u_k)x] = \sum_r C(r-\nu_k) \exp[i2\pi(r-\nu_k)x] \tag{3.14}$$

so that

$$\ell(x) = \sum_{k=1}^{M} w_k \left|1 - h(x) \sum_r C(r-\nu_k) \exp[i2\pi(r-\nu_k)x]\right|^2 \tag{3.15}$$

We now construct the error quantifier ℓ as an integral over all ν_k from 0 to 1, based on Equation 3.15. The only k-dependence on the right is through $\sum_k w_k$, which is just a scale constant. Upon ignoring this data-dependent factor, we have the general dimensionless local error limit

$$\begin{aligned}
\ell(x) &= \int_0^1 \mathrm{d}\nu \left|1 - h(x) \sum_r C(r-\nu) \exp[i2\pi(r-\nu)x]\right|^2 \\
&= 1 - 2h(x) \int_0^1 \mathrm{d}\nu \sum_r C(r-\nu) \cos[2\pi(r-\nu)x] \\
&\quad + h(x)^2 \int_0^1 \mathrm{d}\nu \sum_{r_1} \sum_{r_2} C(r_1-\nu) C(r_2-\nu) \cos[2\pi(r_1-r_2)x]
\end{aligned}$$

We define $\ell(x)$ as the 'map error function' or 'image error function'. Since $\ell(x)$ is dependent on both the gridding function and the correcting function, the choice of these is crucial to obtain a small value of $e(x)$. We take $\ell(x)$ as a quantitative measure of the performance of a given gridding function, in terms of the DFT and FFT dirty image difference. It can be rewritten as:

$$\ell(x) = 1 - 2h(x)c(x) + h(x)^2 \sum_r c(x-r)^2 \tag{3.16}$$

where $c(x)$ is the real part of the inverse Fourier transform of the gridding function $C(u)$, and can be written as:

$$c(x) = \int \mathrm{d}u C(u) \cos(2\pi u x) = \int_0^1 \mathrm{d}\nu \sum_r C(r-\nu) \cos[2\pi(r-\nu)x] \tag{3.17}$$

Assuming for the moment that $C(u)$ is given, Tan (1986) set $\frac{\partial l(x)}{\partial h(x)} = 0$ to find the minimum of $\ell(x)$, giving the optimal $h(x)$ in terms of $c(x)$:

$$h(x) = c(x) / \sum_r c(x-r)^2 \tag{3.18}$$

According to Equation 2.13 in Tan (1986), the sum $\sum_r c(x-r)^2$ can be written as

$$\sum_r c(x-r)^2 = \int_0^1 \mathrm{d}\nu \sum_{r_1} \sum_{r_2} C(r_1-\nu) C(r_2-\nu) \cos[2\pi(r_1-r_2)x] \tag{3.19}$$

This expression is more convenient as it involves only finite sums and an integral over the unit interval. There is 1:1 correspondence between $h(x)$ and $C(u)$, so that $\ell(x)$ can be considered in terms of only $h(x)$ or $C(u)$. This formula for the correcting function differs from that of Greisen (1976), for which $h(x)$ is:

$$h(x) = 1/c(x) \tag{3.20}$$

These two expressions for $h(x)$ will be compared in §3.3.1.

We also note that, as a consequence of Equation 3.16 and 3.18, we have the mathematically correct result $l(x) = 1 - c(x)h(x)$. However, even double precision arithmetic will not suffice to enable its use for $W > 6$.

The function $\ell(x)$ determines the map error levels at certain positions on the image plane. The overall map error may be written as a normalised integral of the map error function $\ell(x)$ across the cropped image plane $-x_0 \leq x \leq x_0$ (where $x_0 \in (0, 0.5]$), as follows:

$$E = \frac{1}{2x_0} \int_{-x_0}^{x_0} \ell(x) \mathrm{d}x \tag{3.21}$$

By minimising E with x_0 fixed, the combination of $h(x)$ and $C(u)$ which minimises the misfit between the DFT and FFT images can be determined. The integral is considered over only a restricted range $[-x_0, x_0]$ because the rest of the dirty image will be discarded to prevent aliasing around the image edge. For example, if we choose $x_0 = 0.25$, as O'Sullivan (1985) and Jackson et al. (1991) suggest, we retain only the central half in each dimension of the dirty image made.

3.2.2 Least–misfit gridding function

The gridding function optimised by minimising the integral map error function E is termed the *least–misfit gridding function*. To perform the optimisation, the least–misfit gridding function $C(u)$ should be substituted by its correcting function $h(x)$, or vice versa. We first follow the analysis of Tan (1986) and optimise E by differentiating with respect to $C(r-\nu)$, denoted by $C_r(\nu)$. Since ν is arbitrary, the integral over ν is irrelevant, and we find a set of simultaneous equations for $C_r(\nu)$:

$$\sum_r \int_{-x_0}^{x_0} \mathrm{d}x h(x)^2 \cos[2\pi(r'-r)x] C_r(\nu) = \int_{-x_0}^{x_0} \mathrm{d}x h(x) \cos[2\pi(r'-\nu)x] \tag{3.22}$$

where the indices have been corrected slightly from Equation 2.24 in Tan (1986). To be explicit, the (Toeplitz) matrix $A_{r'r}$ (which does not depend on ν) is

$$A_{r'r} \equiv \int_{-x_0}^{x_0} \mathrm{d}x h(x)^2 \cos[2\pi(r'-r)x] \tag{3.23}$$

Upon multiplying by the inverse of $A_{r'r}$, we find that

$$C_r(\nu) = \sum_{r'} A^{-1}_{r'r} \int_{-x_0}^{x_0} \mathrm{d}x h(x) \cos[2\pi(r'-\nu)x] \tag{3.24}$$

For any value of ν ($0 < \nu < 1$), this equation delivers W values from the gridding function. In this way, E will be represented only by $h(x)$.

The optimisation procedure is different from that presented by Tan (1986). The variables we aim to optimise are the values of $h(x)$ for a given W on evenly–spaced grids of $N_g + 1$ points, from $x = 0$ to $x = x_0$. From $x = 0$ to $x = x_0$, the function $h(x)$ increases smoothly from $h(0) = 1$. The fucntion $h(x)$ is symmetric about $x = 0$, so that it has a total of $(2N_g + 1)$ points. For the variable ν, we use a grid of M_g equally–spaced points in the interior of the interval $0 < u < 0.5$. As a result of the symmetry of $C(u)$, it has a total of $2WM_g$ points for $C(u)$. These grids need be of only modest size: $N_g = 32$ and $M_g = 16$ suffice for the examples presented here.

We use the following procedure to generate $E(h(x))$:

1) From an initial $h(x)$ of $(N_g + 1)$ values from $x = 0$ to $x = x_0$, we calculate $A_{r'r}$ from Equation (3.23), using a trapezoidal rule to calculate the integral. We then can determine $A^{-1}_{r'r}$.
2) Next, we evaluate $C_r(\nu)$ using Equation (3.24); again we use a trapezoidal rule for the integral on the RHS.
3) We then calculate $c(x)$ from $C_r(\nu)$ using Equation (3.17), and also calculate $h(x)$ based on Equation (3.18). The integral over the ν grid is a simple sum, not a trapezoidal rule.
4) We calculate $\ell(x)$ from Equation (3.16), again as a simple sum over the ν grid.

Table 3.1: Integral of map error function $\ell(x)$ over $x \in [0, 0.25]$ using the least–misfit gridding function, W varying from 1 to 5

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.25$
1	1.61×10^{-2}
2	2.28×10^{-4}
3	1.50×10^{-6}
4	9.18×10^{-9}
5	8.04×10^{-11}

5) We calculate E from Equation (3.21), using the trapezoidal rule for the integral over x. The Levenberg–Marquardt algorithm (Moré 1978) will be used to perform the minimisation of E. We make use of modern `Python` libraries, and the E function is sent to the function `leastsq` from package `scipy.optimise`, which returns the optimised $h(x)$ with $(N_g + 1)$ values.

Once $h(x)$ has been optimised, the function $C(u)$ can be calculated from Equation 3.22 or Equation 3.24 on finer grids, then $h(x)$ from Equation 3.18 and $l(x)$ from Equation 3.16.

In practice, we do not have to form the inverse matrix $\boldsymbol{A}^{-1}$ explicitly, instead, we can use the linear equation solver to solve Equation 3.22. Function `dgels()` in `BLAS` from the package `LAPACK` (**?**) can be used, as it takes an arbitrary number of inputs, which in this case are the RHS of Equation 3.22.

An initial guess for $h(x)$ is necessary to begin the optimisation. Let us introduce the notation $h_W(x)$ to distinguish between correcting functions with differing support widths. The choice $h(x) = 1$ works well if $W \leq 5$. For $W > 5$, the correcting function dips below one for this initial choice of $h(x)$, and the gridding function goes negative even with a significantly large iteration number 2×10^4 during the optimisation process. This unusual phenomenon can be overcome by providing the initial $h_W(x)$ as a function of the optimised $h_{W-1}(x)$:

1) $h_W(x) = 1$ for $W \leq 4$;

2) $h_W(x) = h^2_{(W-1)}(x)/h_{(W-2)}(x)$ for $W > 4$.

According to Equation 3.24 there will, for a given ν, be W values of $C(u)$ calculated. Hence, we can form the least–misfit gridding function by calculating M_g sets of W gridding function values and reordering these according to the corresponding values of $(r - \nu)$. The full gridding function from $-W$ to W has $2WM_g$ discrete values, but only M_g sets of $C_r(u)$ are calculated (with $0 \leq \nu \leq 0.5$) due to the symmetry of the gridding function.

Results of optimisation with W varying from 1 to 5 and $x_0 = 0.25$ are shown in Figure 3.6, with the corresponding integral of the map error function over $x \in [0, 0.25]$ listed in Table 3.1.

There is an obvious discontinuity in the gridding function in Figure 3.6 when $W = 3$. Our least–misfit gridding function is optimised on fine grids, meaning that it is not a continuous

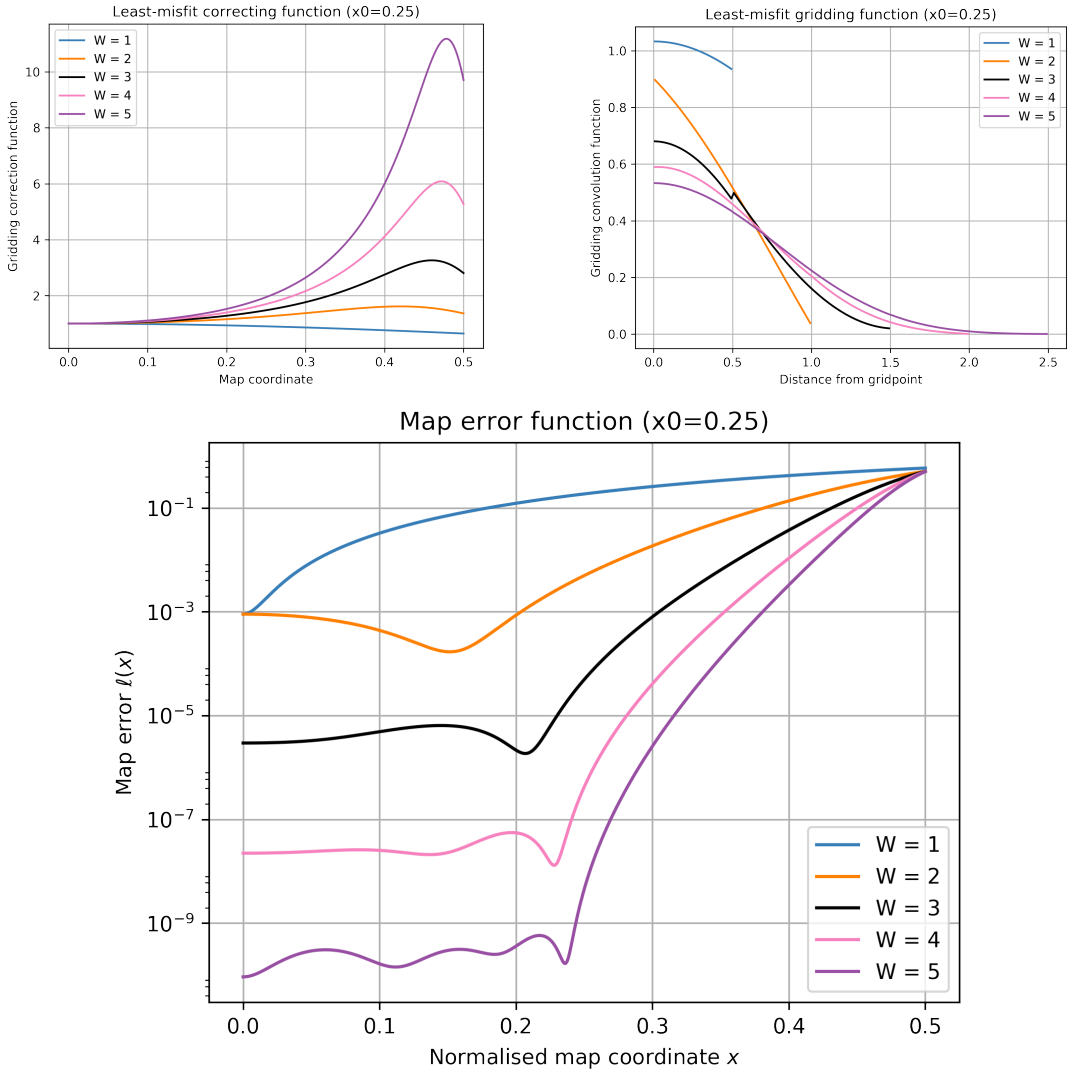


Figure 3.6: The least–misfit gridding functions, and the corresponding correcting functions and map error functions when $x_0 = 0.25$ with W varying from 1 to 5.

function expressed in analytical form. The abrupt change at $u = 0.5$ in the gridding function with $W = 3$ demands explanation.

For $W = 3$ and $\nu = 0.5$, the optimised weights for the corresponding three grid points are respectively 0.020, 0.472 and 0.504, where two of the weights should be identical according to the symmetry of $C(u)$. This asymmetry results in the discontinuity shown. We believe that this exception is related to the relatively large value of $C(1.5) = 0.020$, since the vertical range of the discontinuity in the plot is identical to the value of $C(1.5)$. This would also explain why the visible discontinuity exists only for $W = 3$, since the least–misfit gridding functions for larger Ws already have values near to zero at $u = \pm W/2$.

In further explanation, Figure 3.7 shows a schematic diagram in which the visibility data

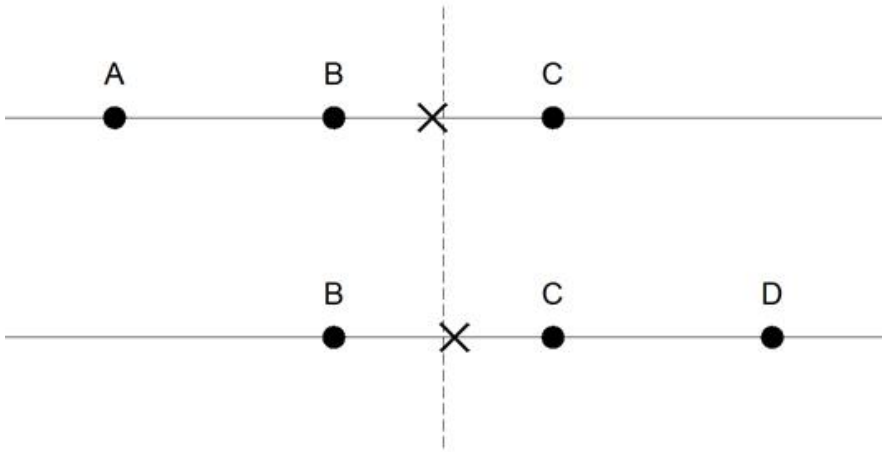


Figure 3.7: Schematic diagram for explaining the discontinuity of the least–misfit gridding function when $W = 3$.

are denoted by crosses, while four close–by grid points marked by 'ABCD' are denoted by solid dots. The vertical dashed line corresponds to $\nu = 0.5$.

When the distance between the visibility and point B is less than 0.5 – for example $\nu = 0.3$ – the visibility will be gridded to grid point A, B and C with weights 0.05, 0.60 and 0.35 respectively. If the distance is greater than 0.5, the visibility will be gridded to B, C and D with the peak weight at C instead. These two examples are consistent with our expectations. When the visibility data falls very close to $\nu = 0.5$, for example $\nu = 0.49$, according to the gridding process shown in Figure 3.4, we expect the peak weight still to be at grid point B, because it is the nearest grid point from the visibility data. It turns out that the weights for grid point A, B and C are 0.02, 0.48 and 0.50 respectively, with the maximum weight at grid point C.

Let us further explain this special case. When B and C have similar distances from the visibility data, they will receive similar weights from the gridding function. However, grid point A happens to have a relatively large weight when $W = 3$, this results in a degradation in B's weight. Similarly, we find that if $\nu = 0.51$, the corresponding weights for B, C and D are respectively 0.50, 0.48 and 0.02, with the maximum weight at grid point B instead of C.

For $x_0 = 0.25$, the optimised least–misfit gridding functions with $W = 6, 7, 8, 9, 10$ and their corresponding correcting functions based on Equation 3.18 are shown in Figure 3.8, in the range $[0, x_0]$. The map error function $\ell(x)$ is also plotted. Since $\ell(x)$ is the upper bound of the square of the image misfit, we can quantify the image misfit by examining values of $\ell(x)$.

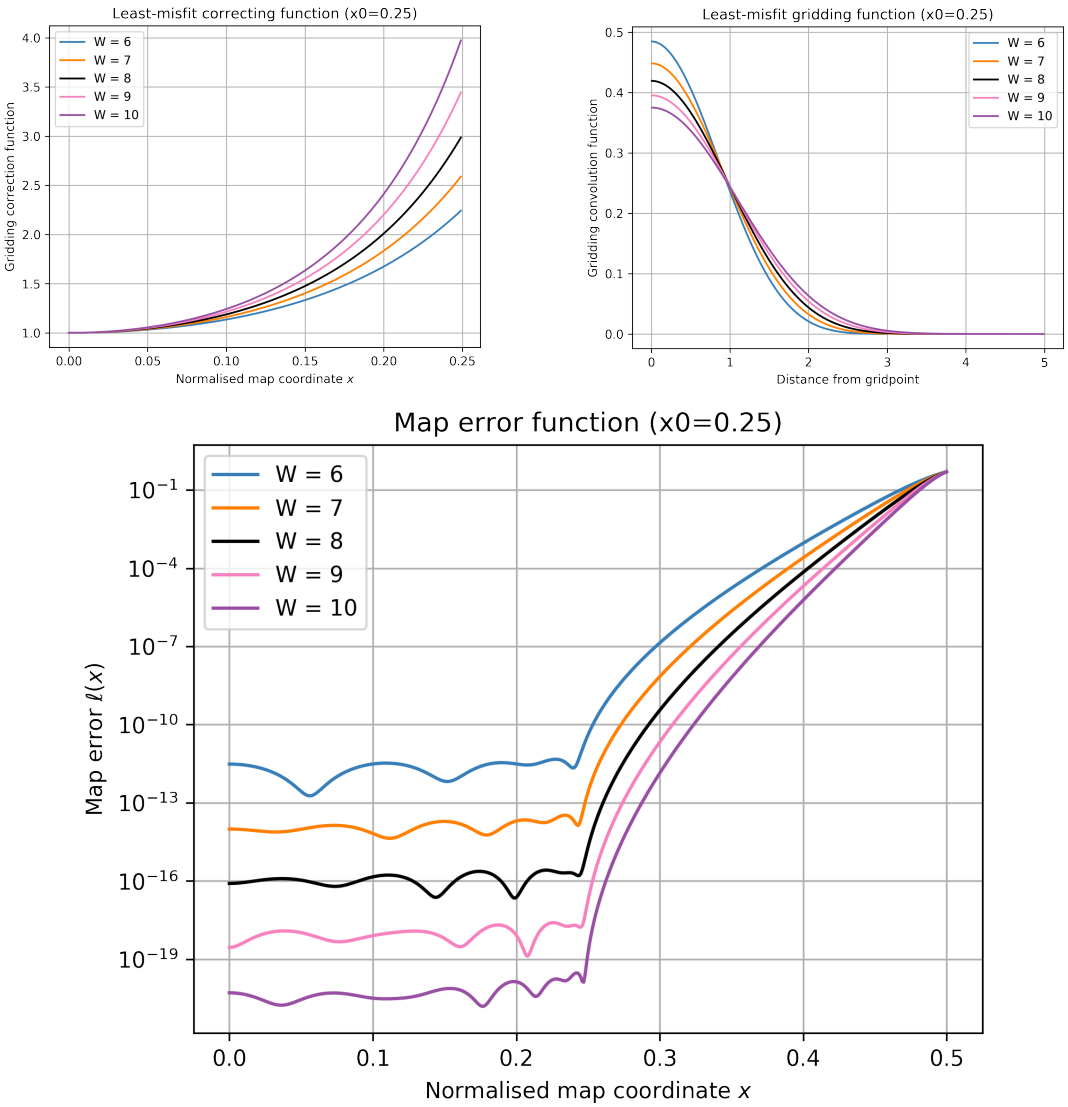


Figure 3.8: The least–misfit gridding functions, and the corresponding correcting functions and map error function $\ell(x)$ with $x_0 = 0.25$ and $W = 6, 7, 8, 9, 10$.

Table 3.2: Integral of the map error function $\ell(x)$ over $x \in [0, 0.25]$ using the least–misfit gridding function with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.25$
6	6.40×10^{-13}
7	3.73×10^{-15}
8	3.02×10^{-17}
9	2.77×10^{-19}
10	2.03×10^{-21}

For each value of W, $\ell(x)$ remains fairly stable, with small fluctuations from $x = 0$ to $x = x_0$

on the normalised map coordinate. When $x > x_0$, the map error function increases rapidly as x increases. This plot indicates that the outer edge of the FFT dirty image ($x \in [x_0, 0.5]$) should be cropped to achieve overall good image misfit. The support width W must be chosen carefully: the larger W, the smaller $\ell(x)$ becomes. Larger values of W will result in higher computational cost of the convolution; will be discussed in Section 3.5.

Table 3.2 displays the integral of the map error function $\ell(x)$ from $x = 0$ to $x = 0.25$ using the least–misfit gridding functions with different values of W. We use double precision during the optimisation process. Upon using a least–misfit gridding function with $x_0 = 0.25$ and $W = 7$, the misfit clearly reaches the limit of single–precision floating–point arithmetic.

Figure 3.9 compares the rational approximation of the gridding function proposed using the same criteria as Tan (1986) against our least–misfit gridding function when $W = 6$ and $x_0 = 0.25$. The plot on the left shows the difference between the two functions, and the plot on the right shows their corresponding map error functions. We observe that the least–misfit function has a error function around 100 times smaller than that of the rational approximation. Since the error function is the upper bound of the square of the image misfit, our updated implementation improves the image misfit approximately tenfold when $W = 6$ and $x_0 = 0.25$.

In Figures 3.10 – 3.13 we use differing values of x_0 and display the optimised least–misfit gridding function with $W = 6, 7, 8, 9, 10$, and the corresponding correcting functions and map error functions. The corresponding integrated map error function are listed in Tables 3.3 – 3.6.

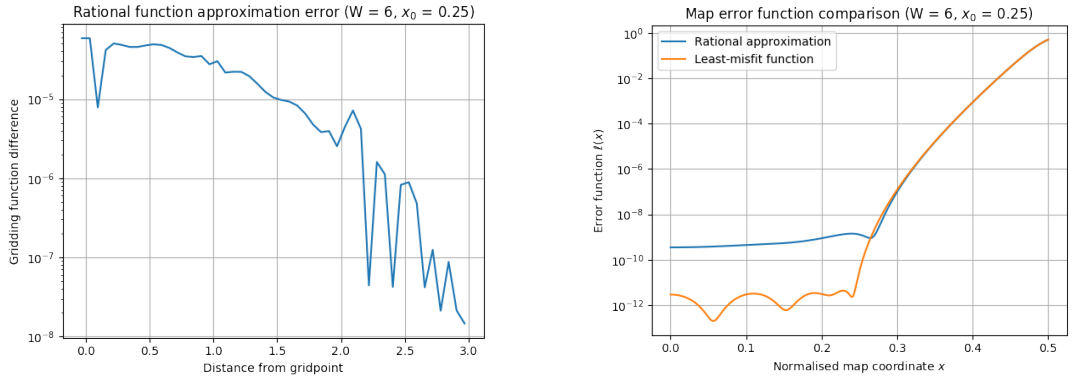


Figure 3.9: The left plot shows the difference between the rational approximation and the least–misfit gridding function with $x_0 = 0.25, W = 6$. The plot on the right shows the corresponding error functions.

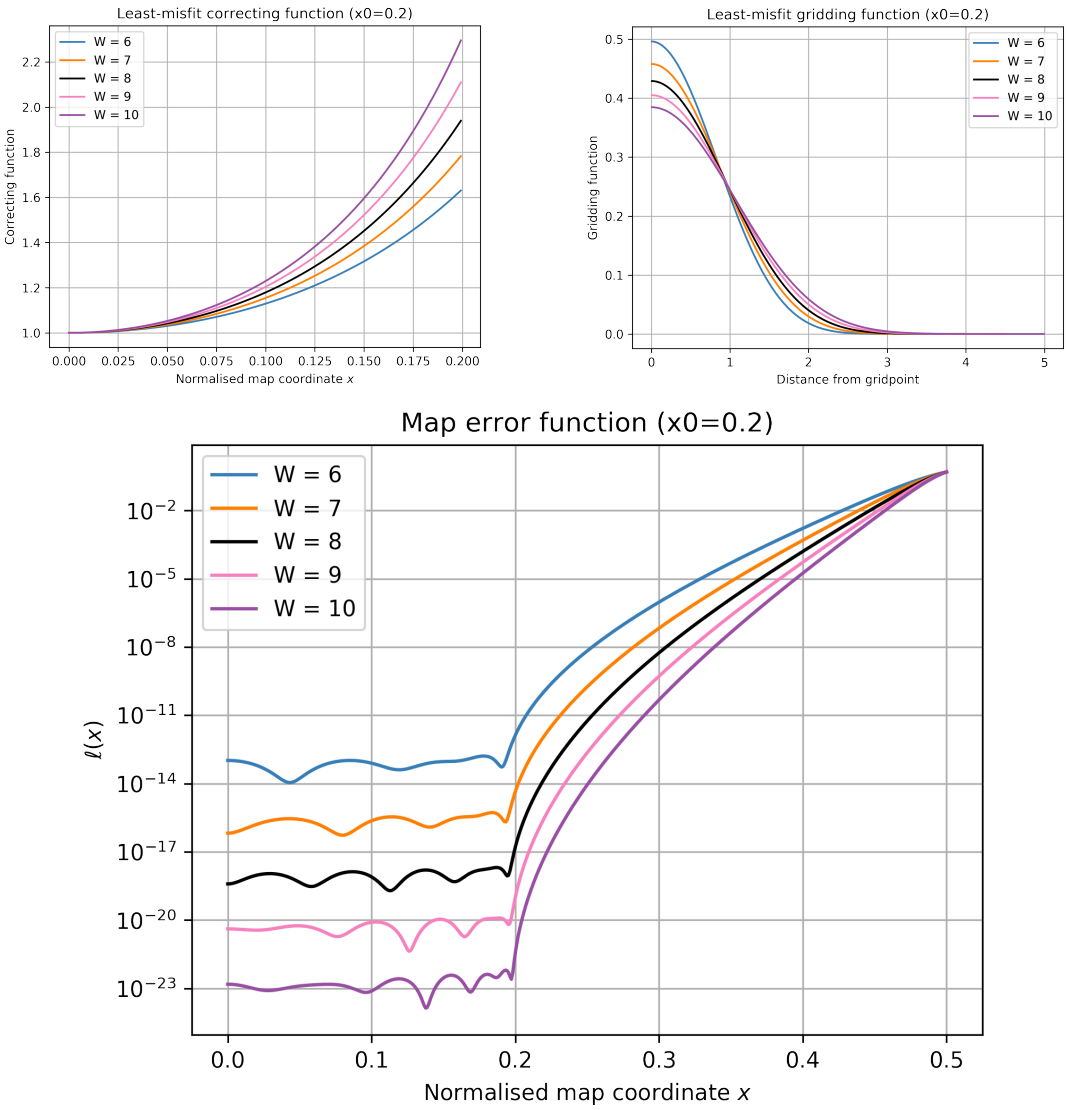


Figure 3.10: The least–misfit gridding function, and the corresponding correcting functions and map error functions for $x_0 = 0.2$ and $W = 6, 7, 8, 9, 10$.

Table 3.3: Integral of the map error function $\ell(x)$ over $x \in [0, 0.20]$ using the least–misfit gridding function with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.2$
6	1.68×10^{-14}
7	4.98×10^{-17}
8	1.94×10^{-19}
9	1.12×10^{-21}
10	3.49×10^{-24}

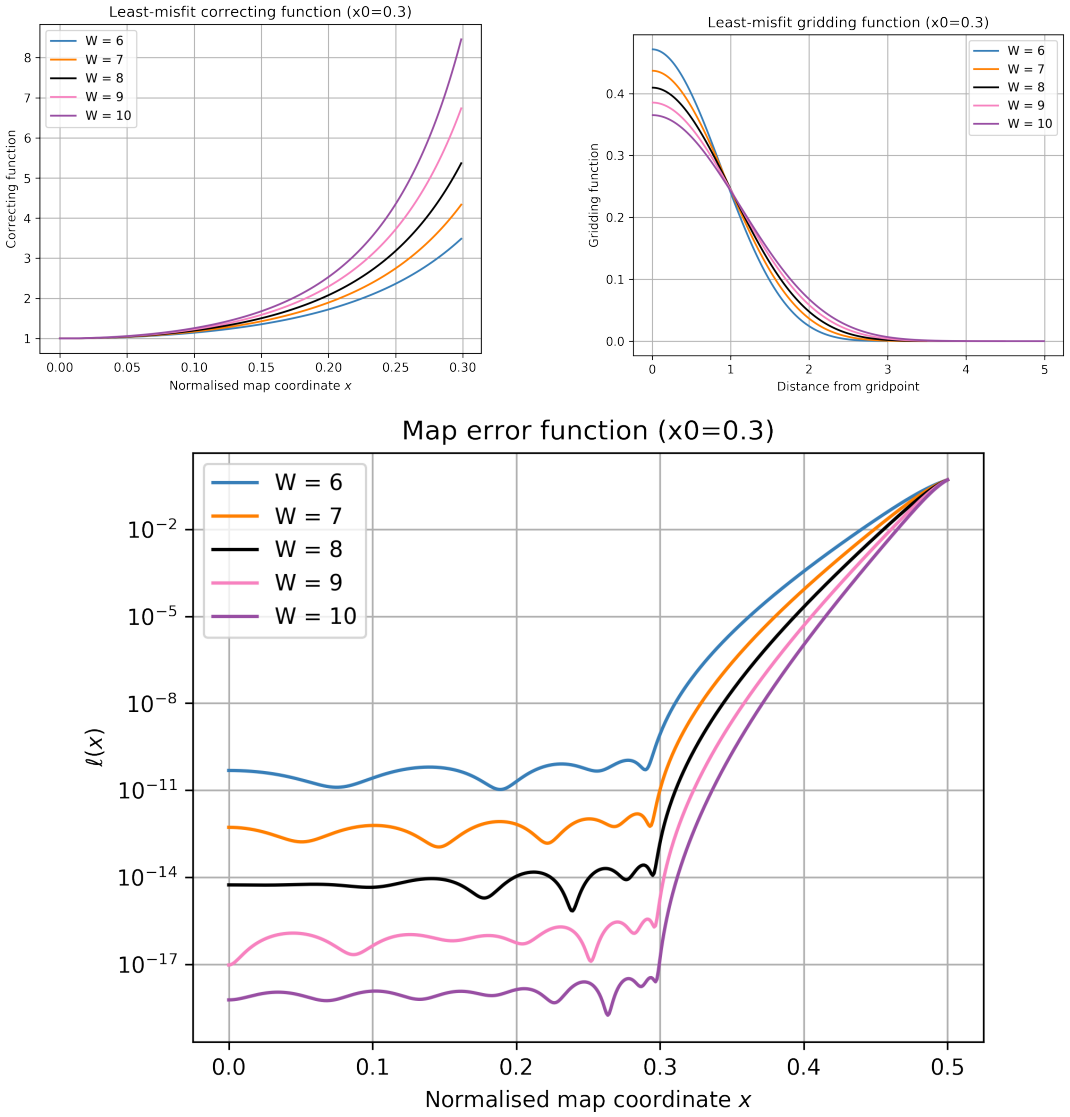


Figure 3.11: The least–misfit gridding function, and the corresponding correcting functions and map error functions with $x_0 = 0.3$ and $W = 6, 7, 8, 9, 10$.

Table 3.4: Integral of the map error function $\ell(x)$ over $x \in [0, 0.30]$ using the least–misfit gridding function with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.3$
6	1.51×10^{-11}
7	1.67×10^{-13}
8	2.53×10^{-15}
9	3.10×10^{-17}
10	3.53×10^{-19}

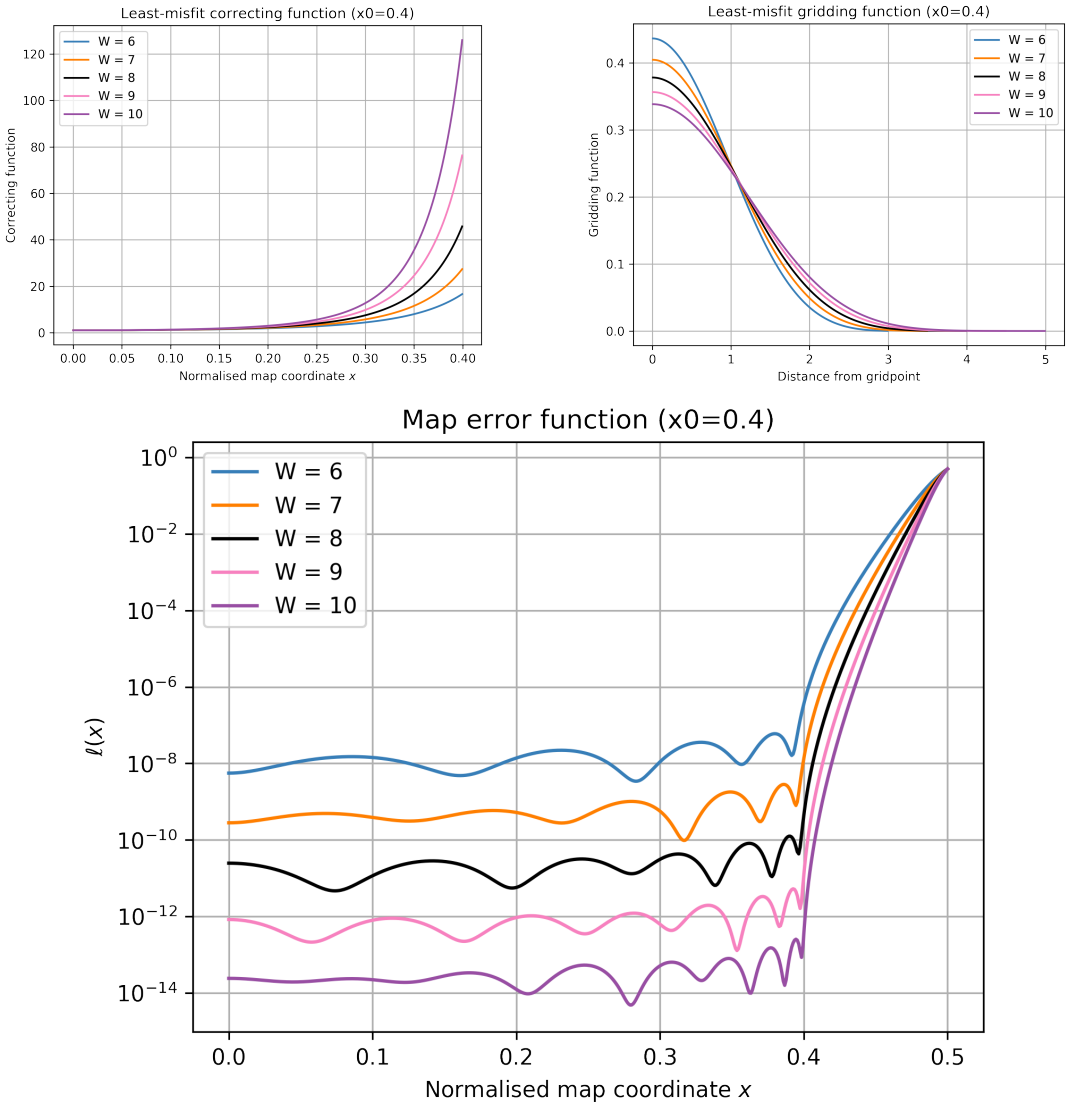


Figure 3.12: The least–misfit gridding function, and the corresponding correcting functions and map error functions with $x_0 = 0.4$ and $W = 6, 7, 8, 9, 10$.

Table 3.5: Integral of the map error function $\ell(x)$ over $x \in [0, 0.40]$ using the least–misfit gridding function with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.4$
6	6.65×10^{-9}
7	2.55×10^{-10}
8	9.79×10^{-12}
9	3.46×10^{-13}
10	1.40×10^{-14}

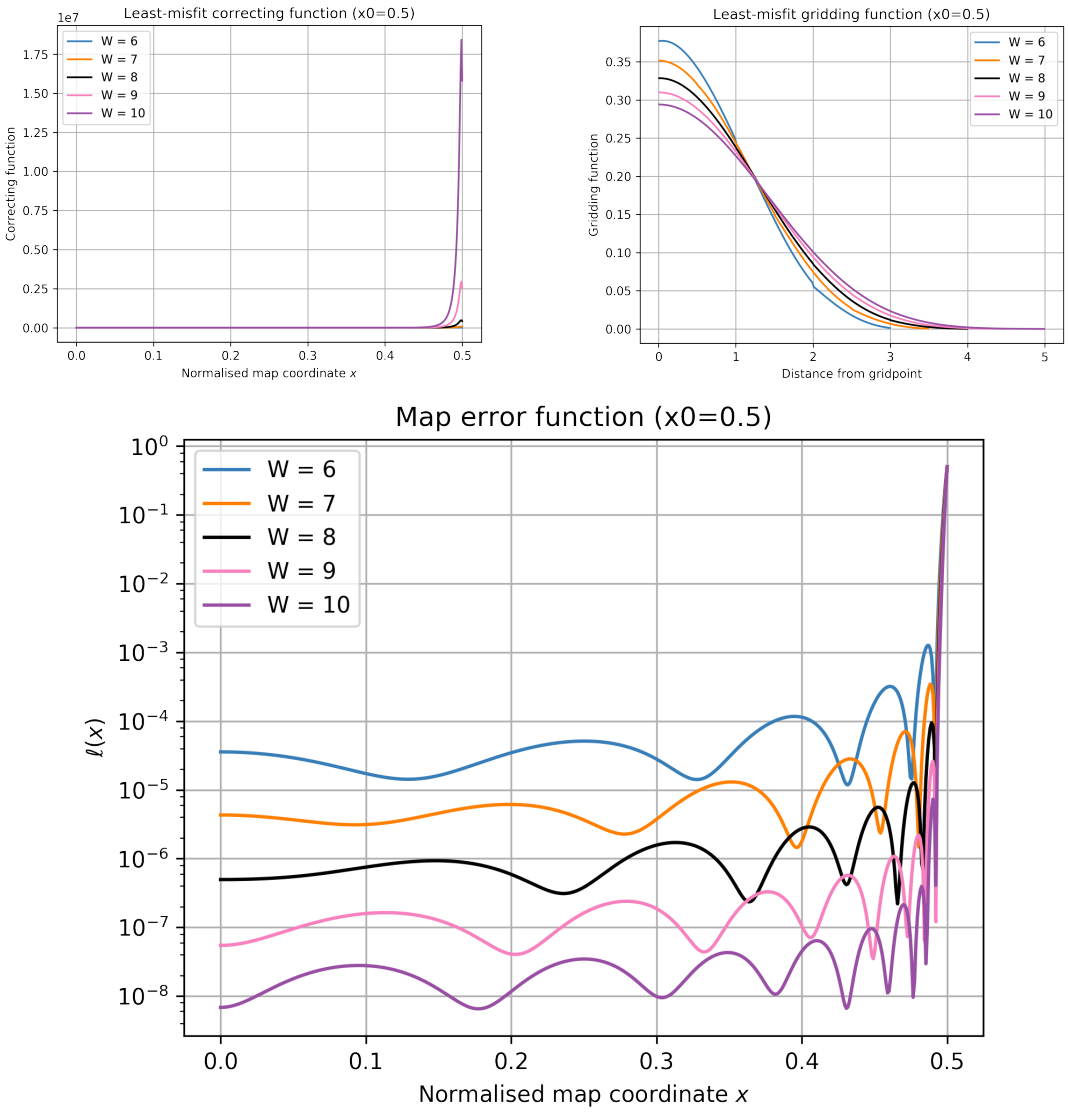


Figure 3.13: The least–misfit gridding function, and the corresponding correcting functions and map error functions with $x_0 = 0.5$ and $W = 6, 7, 8, 9, 10$.

Table 3.6: Integral of the map error function $\ell(x)$ over $x \in [0, 0.50]$ using the least–misfit gridding function with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.5$
6	9.44×10^{-4}
7	7.96×10^{-4}
8	7.08×10^{-4}
9	6.42×10^{-4}
10	5.89×10^{-4}

The map error function $\ell(x)$ optimised with different values of x_0 generally follows the

same trend as in Figure 3.8:

- $\ell(x)$ remains fairly stable, with small fluctuations from $x = 0$ to $x = x_0$, at each value of W.
- As x exceeds x_0, the map error function increases rapidly.
- The larger is W, the smaller the map error function becomes.

We see that the map error function for $x_0 = 0.5$ deteriorates rapidly from very near to 0.5, causing the integrated $\ell(x)$ from $x = 0$ to $x = 0.5$ to be equally bad for different values of W. Consequently, image cropping is still necessary for $x_0 = 0.5$.

Also, the map error function $\ell(x)$ increases as x_0 increases. However, a smaller value of x_0 leads to the cropping of a larger portion of the image, which will result in larger FFT computations to make a larger dirty image. The choice of x_0 must therefore be carefully considered.

3.2.3 Least–misfit gridding function with large support width

Although the optimisation procedure introduced earlier works satisfactorily up to $W = 10$, the matrix $\boldsymbol{A}$ is very badly conditioned. For $W = 10$, there is a factor of about 2×10^6 between its largest and smallest singular values of $\boldsymbol{A}$, as noted in Tan (1986). For $W = 10$ and $x_0 = 0.25$, our implementation of the gridding and FFT procedure described above already yields an accuracy of 10^{-10} compared to DFT[a]. This meets the original desideratum, but let us press further and see how much further the accuracy can be improved.

It is convenient to use $A^{-1}_{rr'}$ as it does not involve ν and can be used repeatedly to calculate $C_r(\nu)$ from Equation 3.24 for as many values of ν as are required. Unfortunately the ill–conditioned matrix $\boldsymbol{A}$ causes the evaluation of $C(u)$ from $h(x)$ to be inaccurate, especially when $W > 12$; see Figure 3.14. The optimisation does not work well for $W > 12$, as the map error functions remain at a similar level even when W increases.

To gain further accuracy it is necessary to reformulate the optimisation problem. Just as the ν integral was irrelevant to the argument leading to Equation 3.22, we can ignore the x-integration and find that the equation for $C_r(\nu)$ now involves a complex 'matrix' $B_{xr}(\nu)$ which depends on the continuous 'index' x:

$$\sum_r B_{xr}(\nu) C_r(\nu) = 1 \tag{3.25}$$

where

$$B_{xr}(\nu) \equiv h(x) \exp[i2\pi(r - \nu)x] \tag{3.26}$$

The right–hand side of Equation 3.25 is a unit complex number for all values of x and ν. Multiplying Equation 3.25 by the transpose of the complex matrix $\boldsymbol{B}^T$ (using the integral over x

[a]The corresponding numerical experiments can be found in §3.4.2

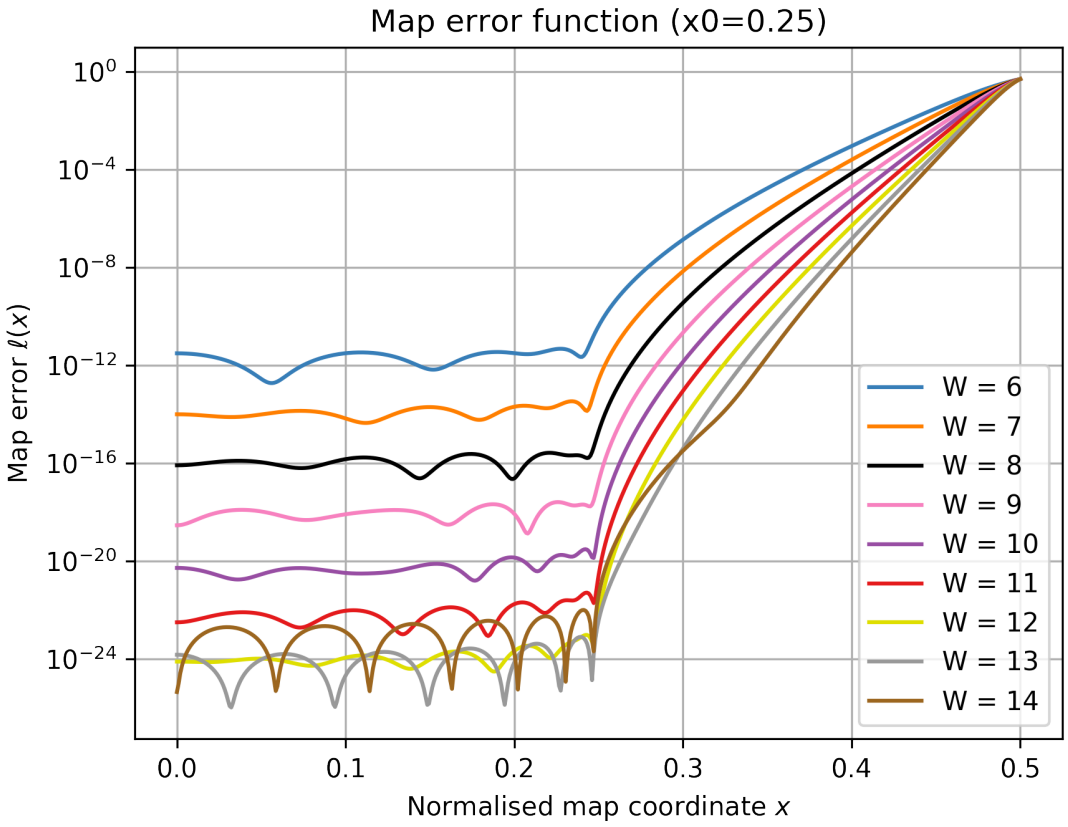


Figure 3.14: The map error functions of least–misfit gridding functions with $x_0 = 0.25$ and W ranging from 6 to 14. The ill–conditioned matrix $A_{r'r}$ causes the evaluation of $C(u)$ from $h(x)$ to inaccurate, especially when $W > 12$.

as the 'matrix' inner product), we observe that any solution of Equation 3.25 is also a solution of Equation 3.22. We also find the relation:[b]

$$\boldsymbol{B}^T \cdot \boldsymbol{B} = \boldsymbol{A} \tag{3.27}$$

The condition number of a matrix $\boldsymbol{B}$ is defined as the ratio of its largest and smallest singular values. The advantage of the present formulation is that the condition number of the $\boldsymbol{B}$ 'matrix', which has at most W non–zero singular values, is the square root of the condition number of the $\boldsymbol{A}$ matrix.

Since each $\boldsymbol{B}$ corresponds to a given ν, we generated 50 uniformly distributed values of ν from 0 to 1 and then calculated 50 corresponding condition numbers. The mean and standard deviation of these are respectively 1.14×10^{-4} and 1.28×10^{-17}. We conclude that the variation of ν has little impact on the condition numbers of $\boldsymbol{B}$. However, the $\boldsymbol{B}$ matrix does dependent on ν, so that the linear equation solver is required for all ν values.

[b] The real part implies Equation 3.22 and the imaginary part vanishes due to the symmetry of $h(x)$.

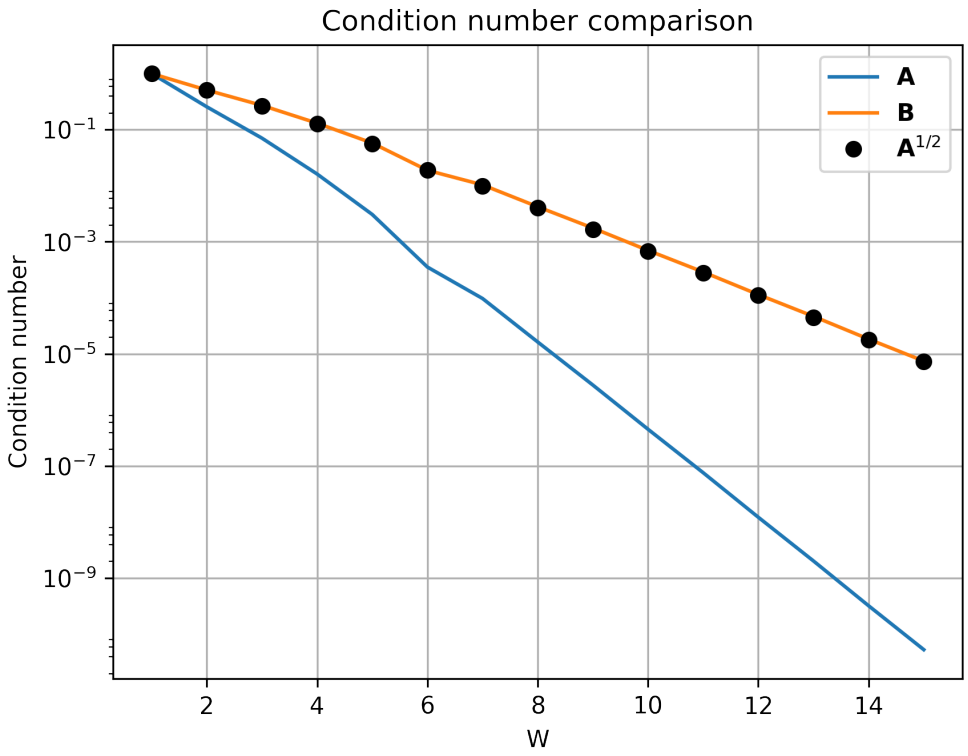


Figure 3.15: Comparison of condition numbers for matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{A}^{1/2}$. The condition numbers of $\boldsymbol{A}$ and $\boldsymbol{B}$ are plotted in blue and orange solid lines respectively against the W values. The condition numbers of $\boldsymbol{A}^{1/2}$ are plotted as black dots, which are the same as the condition numbers of $\boldsymbol{B}$.

In Figure 3.15, we plot the condition numbers of both matrixes, as well as the square root of condition numbers of matrix $\boldsymbol{A}$, with W ranging from 1 to 15. In fact the logarithm of the condition numbers of $\boldsymbol{A}^{1/2}$ plotted in dots are the same as those of $\boldsymbol{B}$, plotted using the orange solid line. This proves our assertion.

In practice, we store the x variable at a discrete set of $N_g + 1$ points from $x = 0$ to $x = x_0$, and our integrals are estimated using a trapezoidal rule on that grid, so that the complex $\boldsymbol{B}$ matrix is just a set of $(2N_g + 1)W$ real numbers for each ν value.

The results of optimisation with W ranging from 6 to 14 have good accuracy, as shown in Figure 3.16. Corresponding integrals of the map error function over $x \in [0, 0.25]$ are listed in Table 3.7.

Upon using a mantissa precision of 160 bits via the `Python` extended precision package `gmpy2` instead of the double precision, we achieve an integrated map error function of 6.62×10^{-43} from $x = 0$ to $x = 0.25$, when using the least–misfit gridding function with $W = 20$ and $x_0 = 0.25$. Figure 3.17 shows the gridding function, its corresponding correction function and

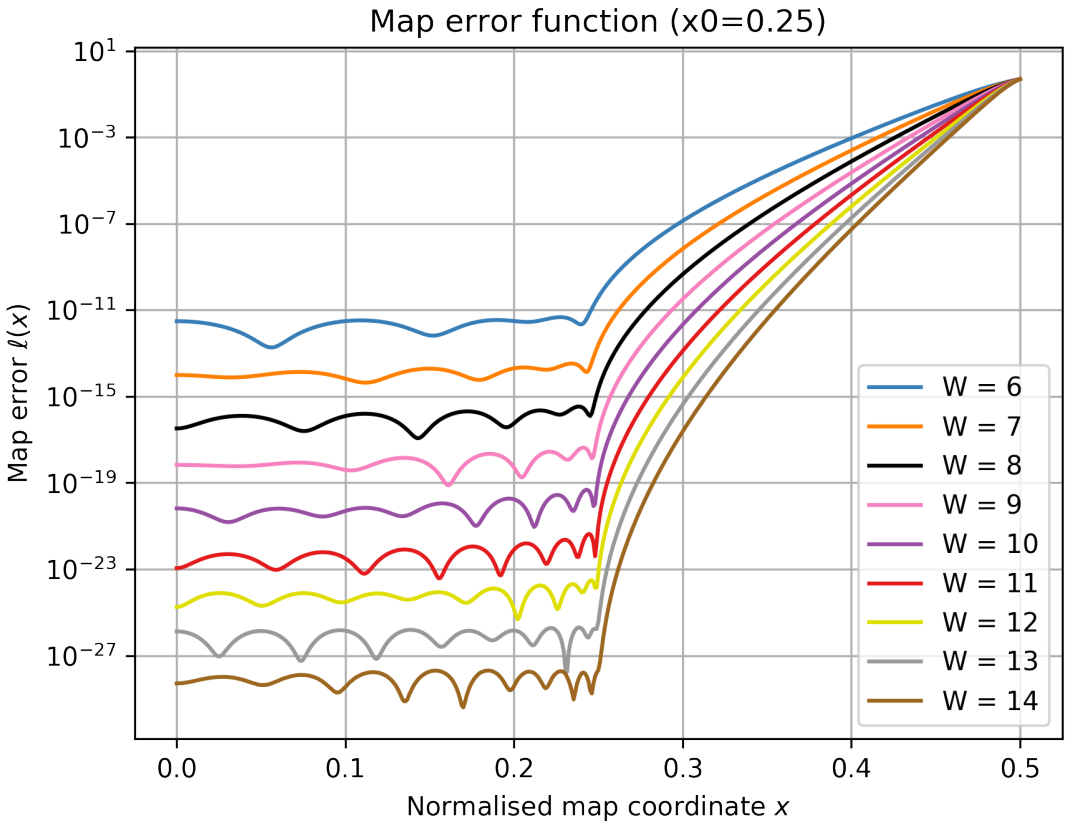


Figure 3.16: Map error functions of the least–misfit gridding functions when $x_0 = 0.25$ with W ranging from 6 to 14.

the map error function $\ell(x)$.

Table 3.7: Integral of the map error function $\ell(x)$ over $x \in [0, 0.25]$ using the least–misfit gridding function ($x_0 = 0.25$) with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.25$
6	6.41×10^{-13}
7	3.63×10^{-15}
8	3.02×10^{-17}
9	2.77×10^{-19}
10	2.03×10^{-21}
11	1.53×10^{-23}
12	1.73×10^{-25}
13	2.14×10^{-27}
14	2.29×10^{-29}

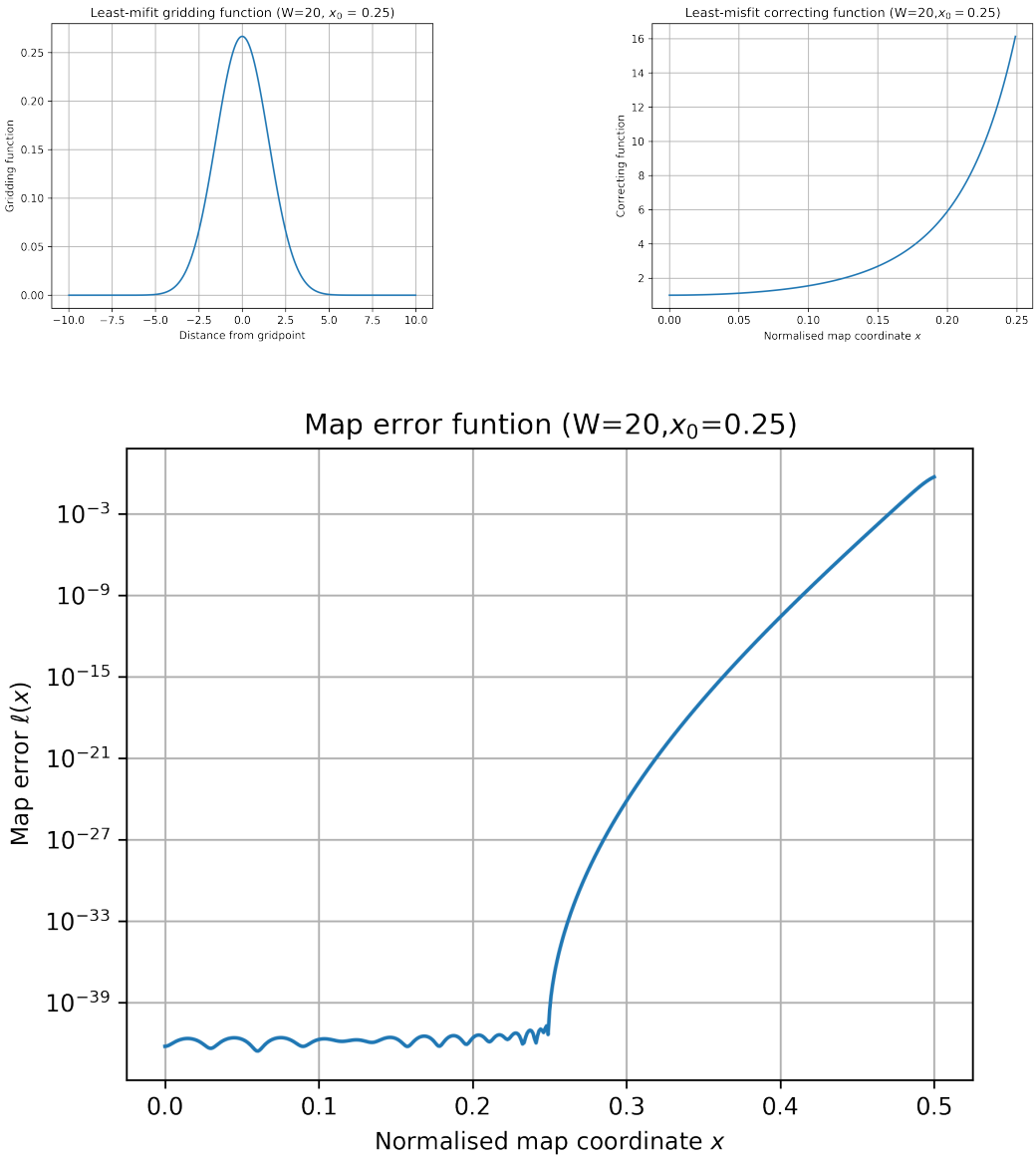


Figure 3.17: The least–misfit gridding function, its corresponding correcting function and the map error function $\ell(x)$ for $x_0 = 0.25$ and $W = 20$.

3.3 Evaluation of spheroidal functions

This section discusses the performance of spheroidal functions. The two different methods of calculating the correcting function are first compared. After that, the error function $\ell(x)$ for the spheroidal function with $\alpha = 1$ is demonstrated. The spheroidal functions ($\alpha = 1$) with different support width are shown in Figure 3.18.

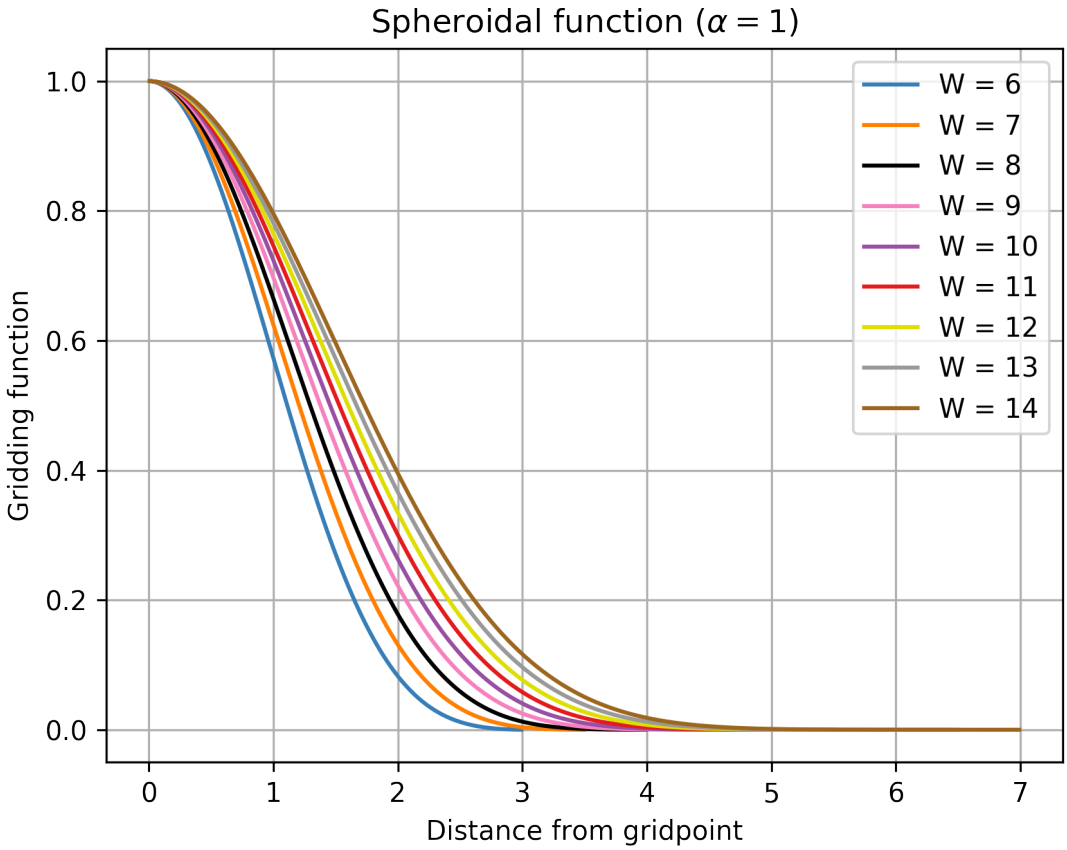


Figure 3.18: Spheroidal functions ($\alpha = 1$) with different support widths.

3.3.1 Correcting function of spheroidal function

The correcting function proposed under the least–misfit criterion in Equation 3.18 is different from the commonly used $\frac{1}{c(x)}$. A study of the two different methods now follows.

The correcting functions of the zero–order spheroidal function with $\alpha = 1$ and $W = 6$ are first calculated using the two different methods, as shown in Figure 3.19. The correcting function calculated based on Equation 3.18 is plotted as the blue line, and that calculated via $\frac{1}{c(x)}$ is plotted as the orange line. In practice we will crop the outer edge of the image, so that only the central fraction of the correcting function is used, where the difference is almost indistinguishable over the part of the image that is retained.

Figure 3.20 displays the difference between the two correcting functions in the range $[-0.5, 0.5]$ and $[-0.25, 0.25]$. The right plot exhibits the difference within the central half of the image; the difference is as little as 10^{-8}. According to Figures 3.19 and 3.20, the difference between the two correcting functions can virtually be neglected in the range $[-0.45, 0.45]$.

Second, we look at the map error function $\ell(x)$ for the spheroidal function with different

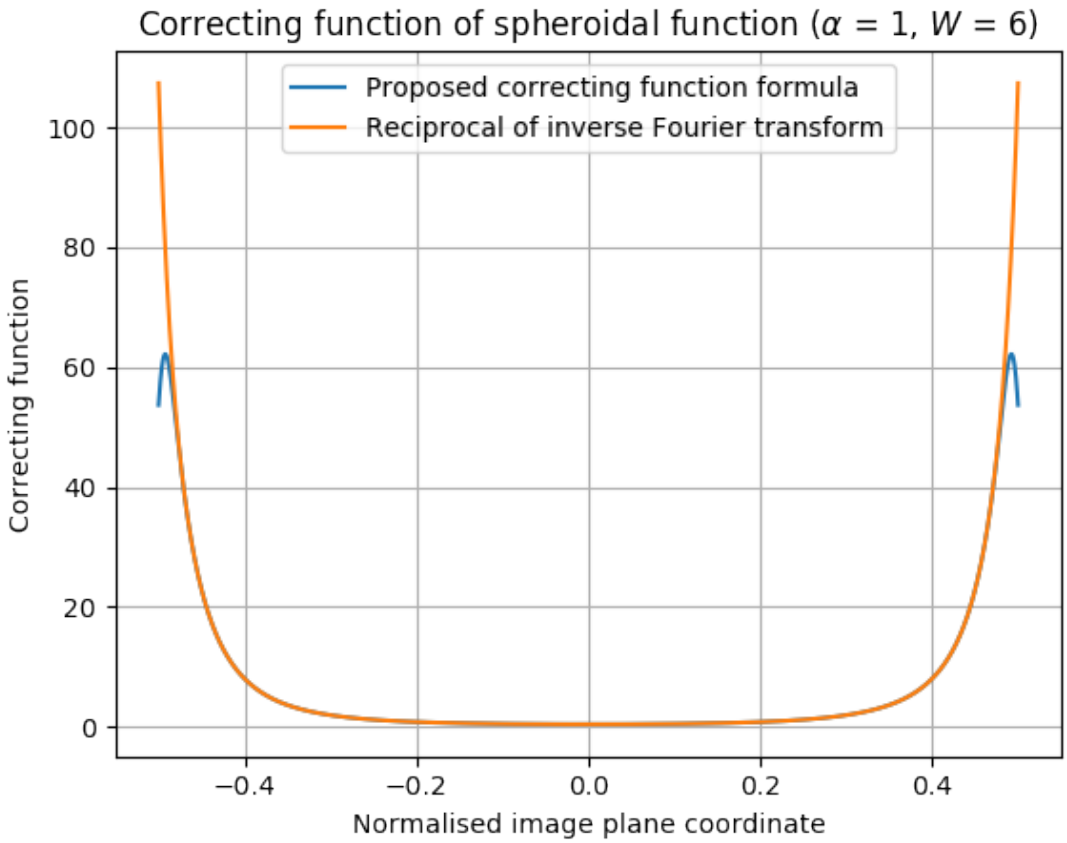


Figure 3.19: Correcting functions for the spheroidal function ($x_0 = 0.25$, $W = 6$) calculated via Equation 3.18 and from $1/c(x)$. The difference inside the central part of the image is almost indistinguishable. In practice, only the central portion of the correcting function will be used.

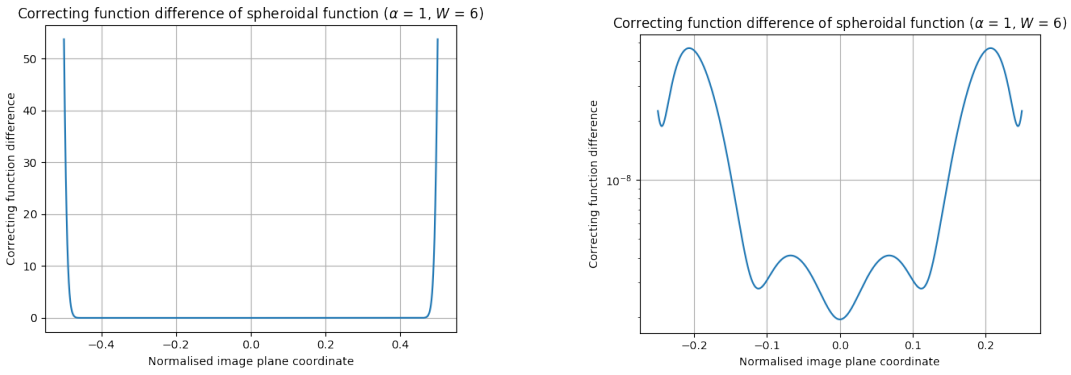


Figure 3.20: Difference between the two correcting functions calculated using different methods. The left plot shows the range $[-0.5, 0.5]$, and the right plot shows the range $[-0.25, 0.25]$.

correcting functions, which are plotted in the top graph in Figure 3.21. The orange line demonstrates the map error function when using $1/c(x)$, and the blue line shows the map error function with the proposed correcting function formula. The difference between these two map error functions is barely distinguishable except for the positions at the image edge. The lower plot shows the difference between the two map error functions. From $x = 0$ to $x = 0.15$, the difference is near to the accuracy limit of double precision. In the range $[0, 0.4]$, the difference is less than 10^{-9}. We conclude that there is no significant difference between the two ways of calculating the correcting function within the retained image range, in terms of the map error function. In the rest of this chapter, we use Equation 3.18 to calculate the correcting functions. The corresponding correcting functions are shown in Figure 3.22.

3.3.2 Map error function of the spheroidal function

The map error functions $\ell(x)$ of the spheroidal function with $\alpha = 1$ are plotted in Figure 3.23. For each value of W, $\ell(x)$ increases steadily with the normalised map coordinate x. Image cropping is vital for spheroidal functions, as the map error function becomes quite large near the image edge. Table 3.8 displays the integrated Map Error E function from $x = 0$ to $x = 0.25$, along with different values of W.

Table 3.8: Integral of the map error function $\ell(x)$ over $x \in [0, 0.25]$ using spheroidal functions ($\alpha = 1$) with different values of W

W	Integrated $\ell(x)$ from $x = 0$ to $x = 0.25$
6	4.88×10^{-9}
7	3.60×10^{-10}
8	1.99×10^{-11}
9	9.06×10^{-13}
10	7.77×10^{-14}
11	2.44×10^{-15}
12	2.86×10^{-16}
13	9.30×10^{-18}
14	9.99×10^{-19}

3.4 Comparison of least–misfit function and spheroidal function

The proposed least–misfit function will be compared with the spheroidal function with $\alpha = 1$, taking into account three factors:

- misfit between the DFT and FFT dirty images,
- aliasing suppression performance
- computational cost.

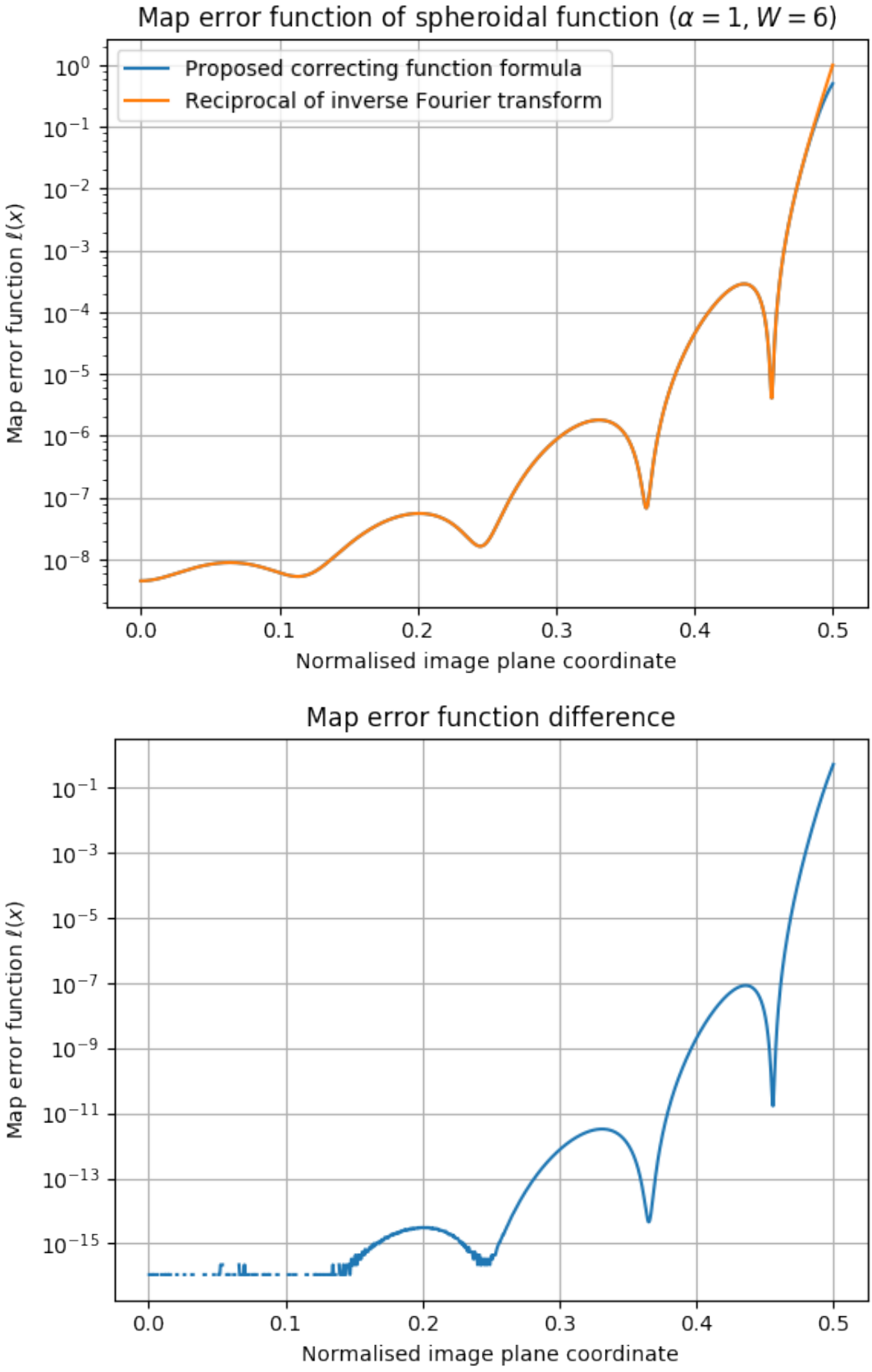


Figure 3.21: Map error functions of the spheroidal function ($\alpha = 1$, $W = 6$) using two different correcting functions are shown in the upper plot. The difference is scarcely distinguishable except for positions around the image edge. The lower plot displays the difference between the two map error functions.

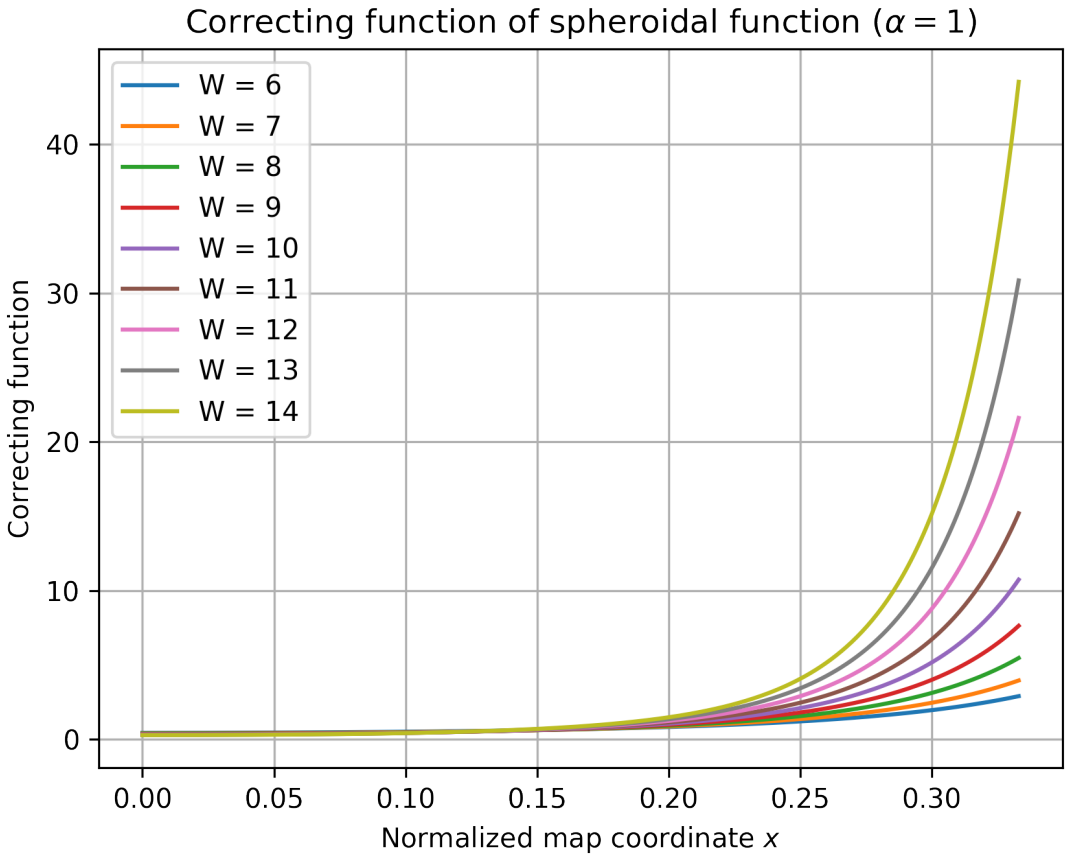


Figure 3.22: Correcting functions for the spheroidal function ($\alpha = 1$) with different support widths.

The present section focuses on the first two factors. We first undertake a theoretical comparison, including comparison of the map error function and the aliasing suppression ratio. After that, a numerical experiment is run to make comparison using simulated data. Computational costs are discussed and compared in Section 3.5.

3.4.1 Theoretical comparison

3.4.1.1 Comparison on theoretical image misfit between DFT and FFT

The theoretical misfit between DFT and FFT dirty images can be measured by the map error function $\ell(x)$. Upon comparing Figure 3.23 to Figure 3.16, it can be seen that the map error functions for the spheroidal functions are at least 10^4 times larger than the least–misfit gridding functions, supposing the same value of W.

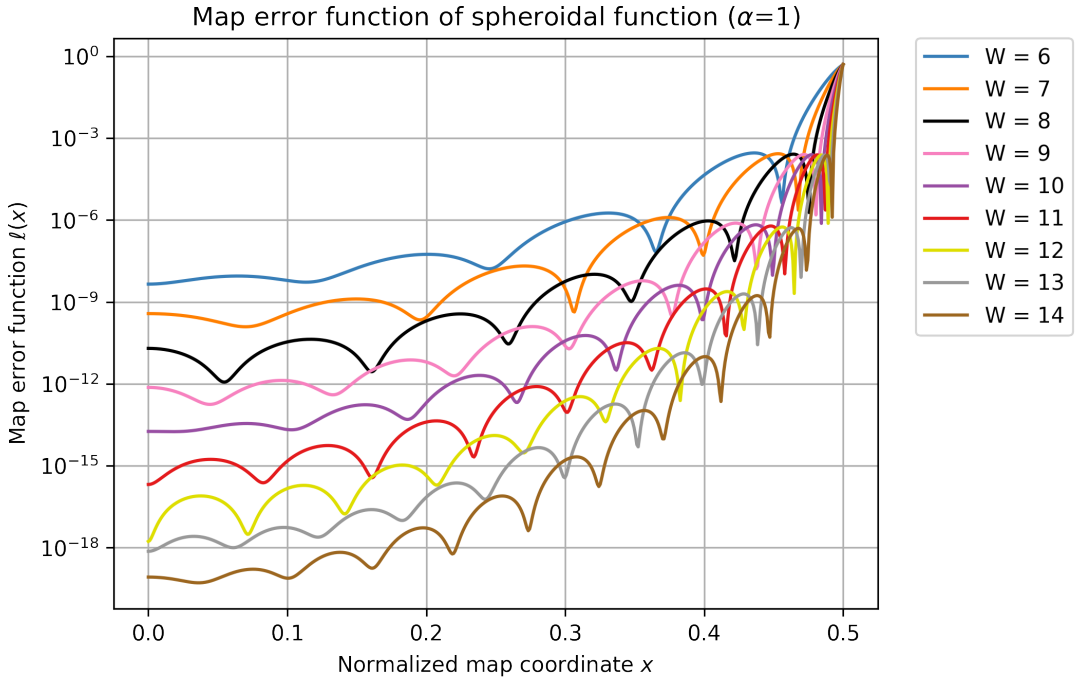


Figure 3.23: The map error functions of spheroidal functions with $\alpha = 1$, and W ranging from 6 to 14.

3.4.1.2 Comparison of aliasing suppression ratio

Aliasing suppression performance is a further factor when choosing the ideal gridding function. The aliasing suppression ratio defined in Equation 3.4 will be used again here to demonstrate the aliasing suppression performance of the least–misfit gridding functions with different values of W and x_0.

Figures 3.24 – 3.28 show the aliasing suppression ratios of the least–misfit functions for different values of x_0 and W. The x–axis is the normalised image plane coordinate, with $x = 0$ representing the image centre and $x = 0.5$ the image edge. Since the image will be cropped to retain only the centre $|x| < x_0$, the values of the suppression ratios at half integers on the x–axis need not be as small as at integer numbers.

We compare these figures for differing values of x_0 and W, and draw the following conclusions:

- When W is fixed, for an increase in x_0, the suppression ratios take larger values at the same position, and wider 'valleys' around integers.
- When x_0 is fixed, for an increase in W, the suppression ratios follow the same pattern but achieve even smaller ratio values around integers.
- For fixed W and x_0, the ratio remains at the lowest value within a relatively wide range around integer numbers, showing a consistently strong capability for aliasing–suppression

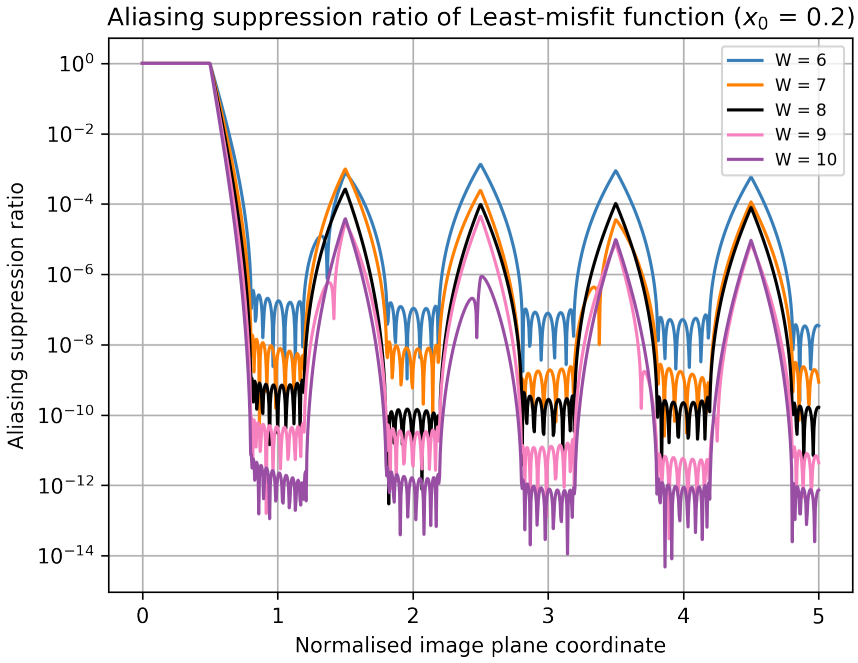


Figure 3.24: The aliasing suppression ratio plot of the least–misfit functions with $x_0 = 0.20$ and $W = 6, 7, 8, 9, 10$.

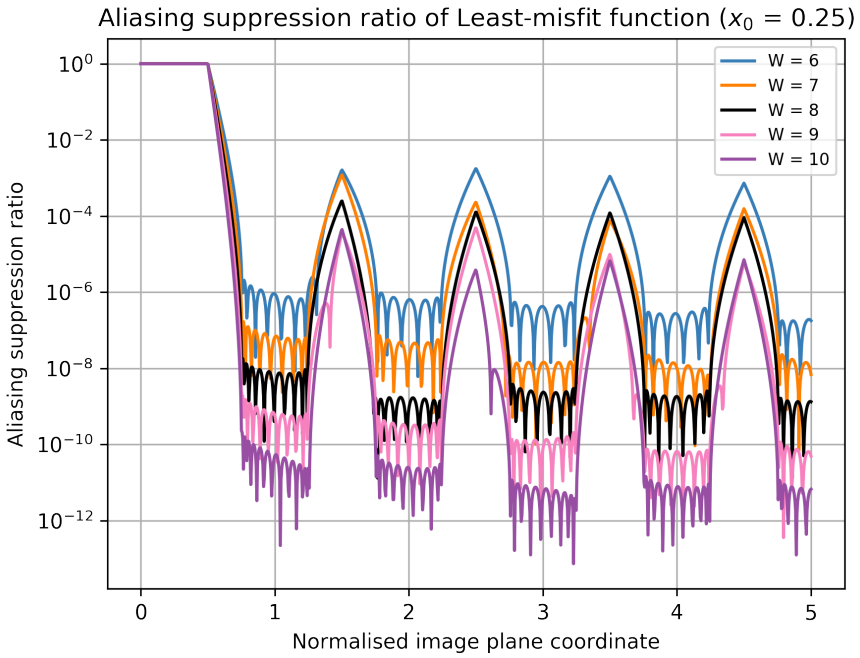


Figure 3.25: The aliasing suppression ratio plot of the least–misfit functions with $x_0 = 0.25$ and $W = 6, 7, 8, 9, 10$.

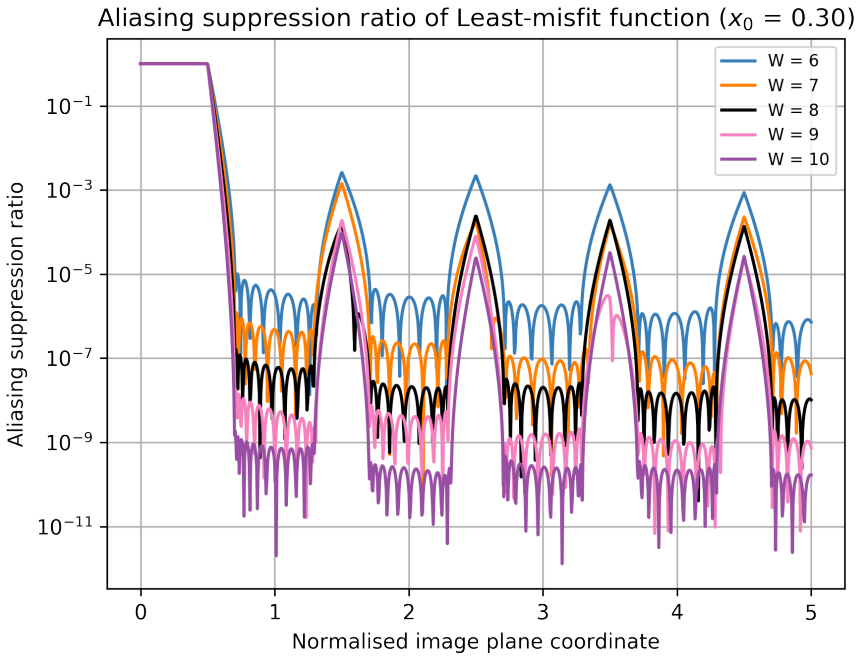


Figure 3.26: The aliasing suppression ratio plot of the least–misfit functions with $x_0 = 0.30$ and $W = 6, 7, 8, 9, 10$.

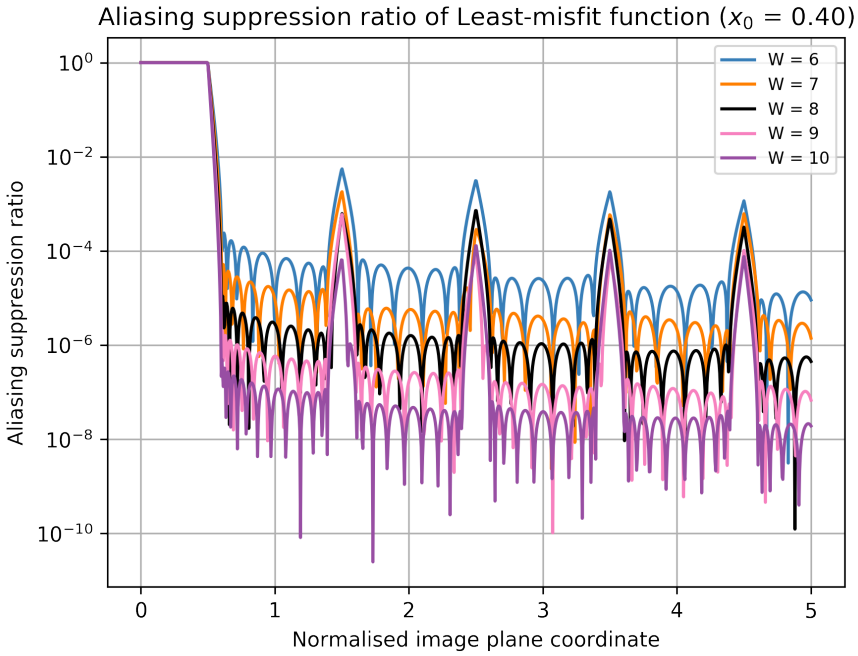


Figure 3.27: The aliasing suppression ratio plot of the least–misfit functions with $x_0 = 0.40$ and $W = 6, 7, 8, 9, 10$.

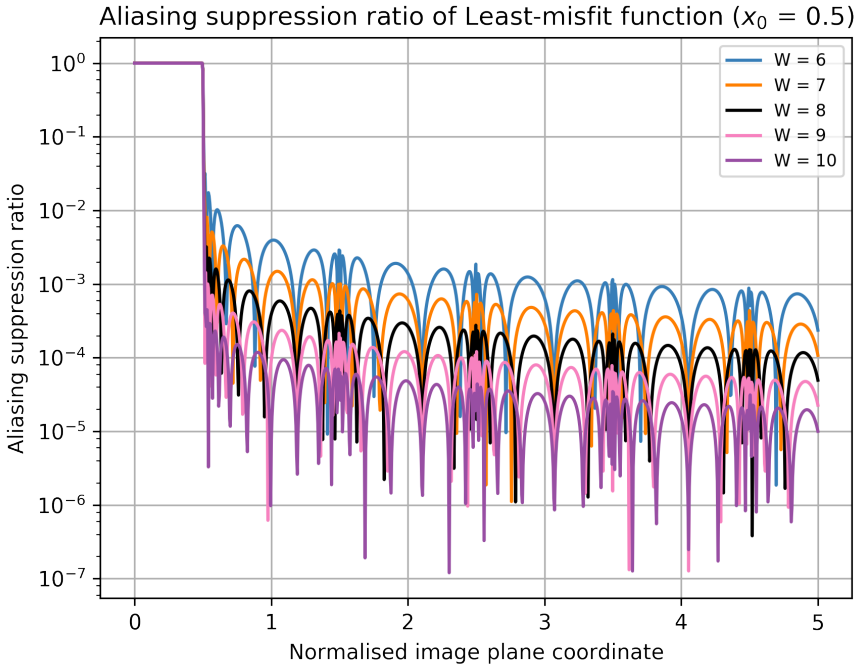


Figure 3.28: The aliasing suppression ratio plot of the least–misfit functions with $x_0 = 0.50$ and $W = 6, 7, 8, 9, 10$.

in areas which can be aliased into the image centre.

The optimised least–misfit function maintains the aliasing suppression ratio at a relatively small level within the areas which can alias into the image centre $[-x_0, x_0]$. These areas correspond to the valleys shown in the plot. Larger x_0 values require less image cropping, resulting in wider valleys. However, the suppression ratio increases with x_0, indicating worse suppression performance overall.

The second conclusion above indicates that a larger support width gives better suppression of aliasing. These aliasing suppression ratio plots are consistent with the map error function plots seen in Figure 3.10 – 3.13.

Upon comparing aliasing suppression ratios of least–misfit functions to spheroidal functions in Figure 3.5, it becomes clear that least–misfit functions have overall smaller suppression ratios with the same W, indicating better aliasing–suppression ability. To achieve an aliasing suppression ratio no larger than 10^{-7}, it is necessary to choose $x_0 = 0.25$ and $W = 7$ for the least–misfit function. If the spheroidal function is adopted, the same performance can only be achieved by choosing at least $W = 10$ and then cropping the image from at least $x = 0.25$.

Last but not least, the aliasing suppression ratio of the least–misfit function does not show a decreasing trend, in contrast to the spheroidal function. This helps to maintain good aliasing suppression performance across the entire field using the least–misfit function.

3.4.2 Experimental comparison

Based on the discussions in §3.4.1, in terms of the map error function and the aliasing suppression ratio, the least–misfit gridding function with $x_0 = 0.25$ and $W = 10$ already outperforms the spheroidal function ($\alpha = 1$) with W ranging from 6 to 14. In the present section we compare the least–misfit gridding function ($x_0 = 0.25$ is fixed) with the spheroidal function ($\alpha = 1$) by examining the image misfit and aliasing effects based on the same simulated data.

3.4.2.1 Image misfit comparison on simulated one–source data

We begin with a simple one point source snapshot. The point source was simulated based on the VLA A–array configuration in `CASA` using function `simobserve`; it is located at the phase centre $(0, 0)$ and is observed at 4 GHz with a flux of 2 Jy. No noise was added to this simulated data. The DFT dirty image of the data is shown in the top left figure in Figure 3.29, and the FFT dirty image using the least–misfit gridding function with $W = 7$ is shown at top right. The pixel size is 0.02 arcsec, with an image size of 1000 pixels by 1000 pixels. The image misfit is shown at bottom left, with its central half zoomed in as shown in the bottom right figure.

In the bottom two figures, the central half of the image misfit has magnitude 10^{-8}, leaving the difference between the DFT and FFT dirty image in the range of $[-0.25, 0.25]$ as merely rounding errors. The image misfit around $x = \pm - 0.5$ is comparably significant.

To quantify the difference between the DFT and FFT dirty images, or the image misfit, we calculate the RMS of the misfit values within the range $[-x_c, x_c]$ on the normalised map coordinate, $x_c \in (0, 0.5]$. The RMS values can therefore be plotted against the normalised image plane coordinate x. If $x_c = 0.5$, the calculated RMS will be the RMS value for the entire image range; if $x_c = x_0$, only the central half image misfit will be calculated.

Figure 3.30 shows the RMS value of the image misfit versus the normalised image coordinate for the simulated single–source snapshot data. The gridding functions used are least–misfit functions ($x_0 = 0.25$) with different values of W. The x–axis represents the normalised image plane coordinate, and the y–axis represents the image misfit RMS value.

The trend in Figure 3.30 matches what is seen in Figure 3.8. The RMS value of the image misfit remains at a steady small value from 0 to 0.25. It then undergoes a rapid increase from 0.25 to 0.5, where the image will be cropped. The final FFT image therefore maintains a small image misfit from its corresponding DFT image across the whole image. For this specific simulated data, the least–misfit function with $W = 6$ already causes the the image misfit to reach the limit of single precision floating point. The mild fluctuations visible in Figure 3.8 do not manifest significantly here because of the averaging operation in taking the RMS value.

In Figure 3.31 we show the RMS value of the image misfit using the spheroidal function

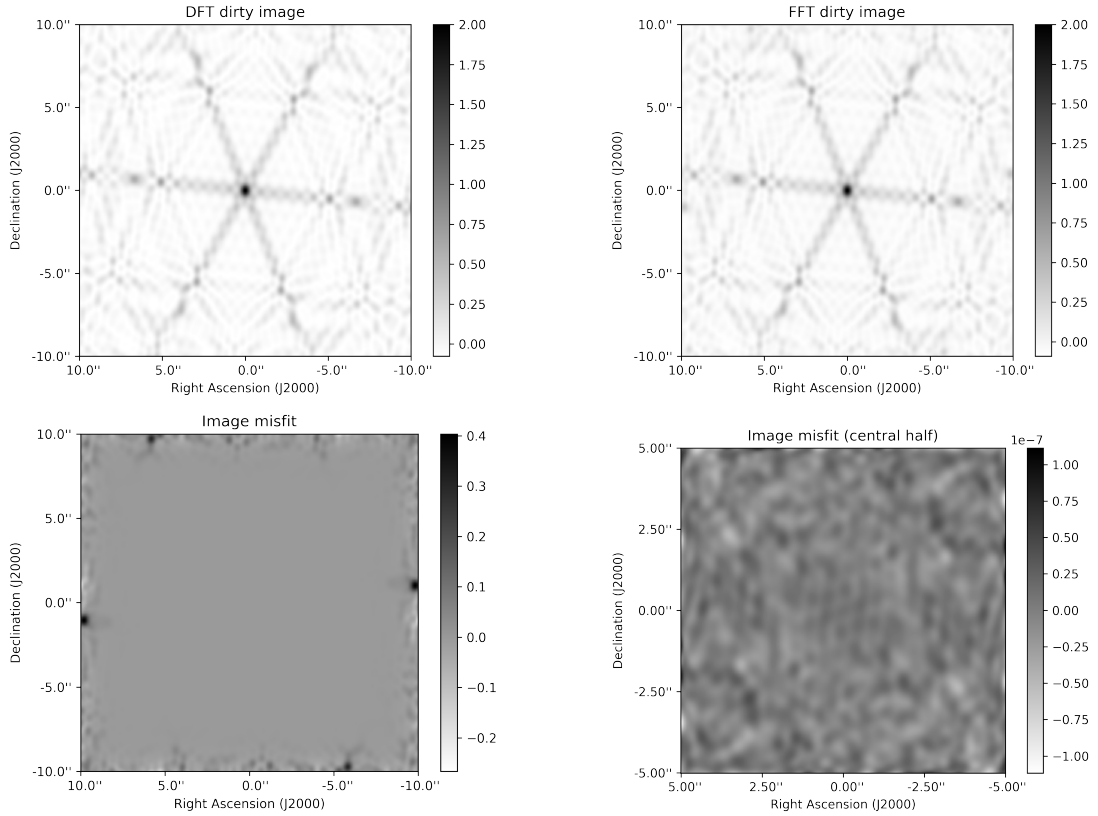


Figure 3.29: DFT dirty image (top left), FFT dirty image using least–misfit gridding function with $W = 7$ (top right), and their difference (bottom left). The bottom right image is the central half of the image misfit. The dirty images are made from the simulated single–source snapshot.

with different Ws and the same data. The trend in this figure matches what is seen in Figure 3.23. The image misfit increases monotonically from the image centre to the edge. For this specific simulated dataset, only for the spheroidal function with $W \geq 8$ does the image misfit achieve the single precision limit in the range $(0, 0.25]$.

Upon comparing Figure 3.30 with Figure 3.31 within $[0, 0.25]$, we find that the image misfit using the least–misfit gridding function is at least 10^2 times better than when using the spheroidal function with the same window width W. This trend is consistent with our theoretical results on map error functions.

In addition, we can make a dirty image using the same data in `CASA` by using the `clean` command with `iter`= 0[c]. We make one dirty image of size 1000 by 1000 through `CASA` for comparison with the DFT dirty image. We also make another dirty image of size 2000 by 2000 via `CASA` with the same pixel size and then crop the outer half of the image, to see whether the image cropping will affect the image misfit.

[c]We found that the dirty images made by `CASA 4.6.0` and the newest `CASA 5.4.0` version are identical. The `clean` command and the `tclean` command also give identical dirty images.

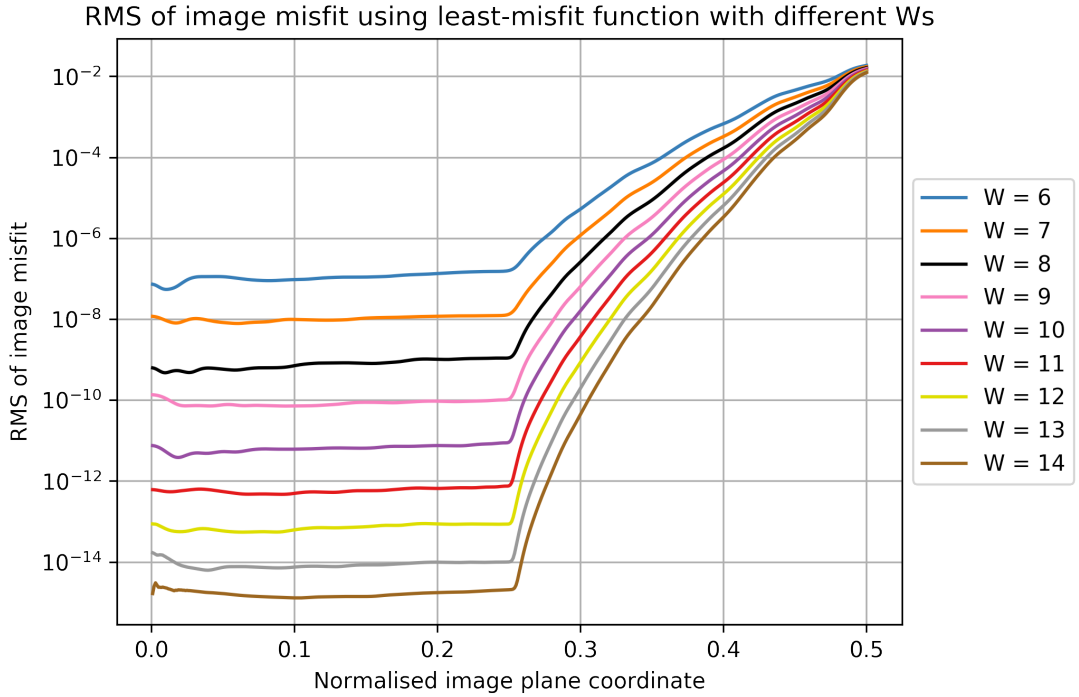


Figure 3.30: RMS value of the image misfit for the single point–source simulated data. The gridding functions used are least–misfit functions ($x_0 = 0.25$) with different values of W.

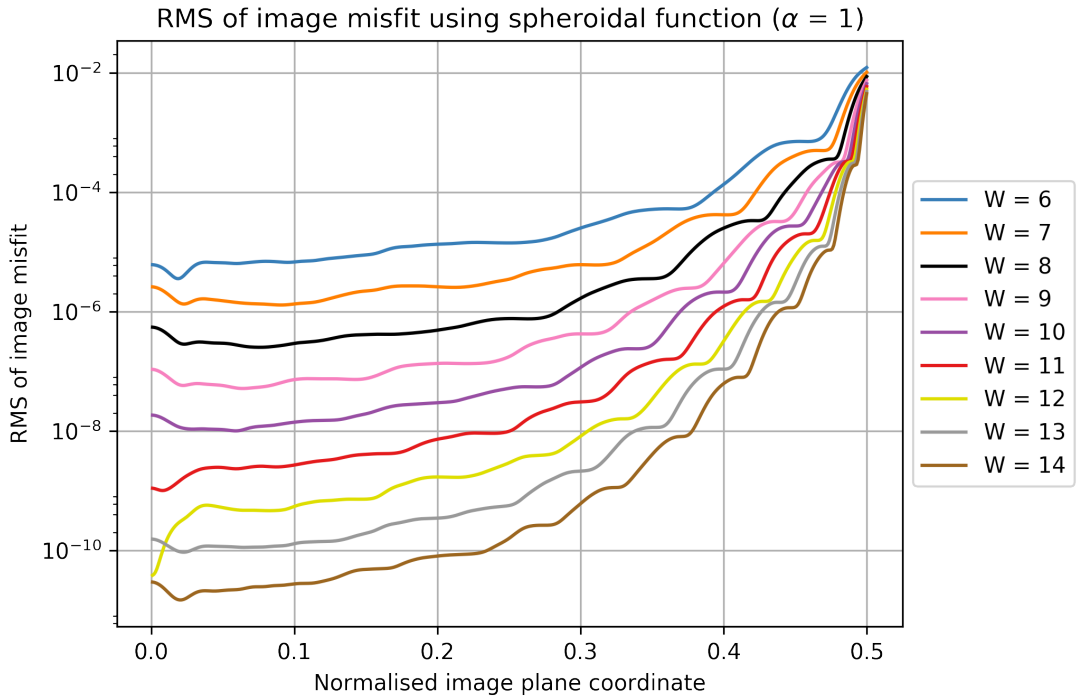


Figure 3.31: RMS of the image misfit on the single point–source simulated data. The gridding functions used during the gridding process are spheroidal functions with different values of W.

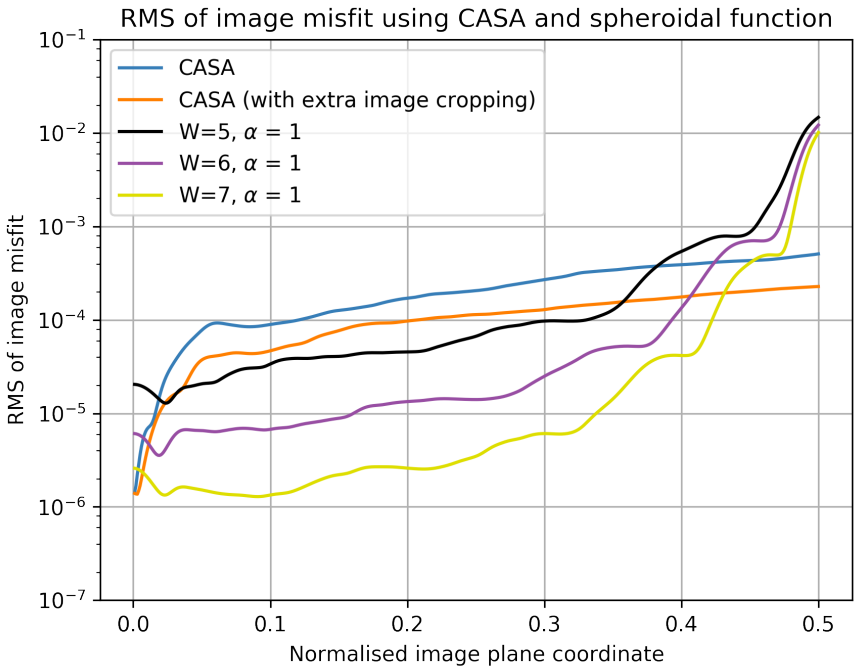


Figure 3.32: RMS value of the image misfit for the single point–source simulated data. The blue line shows the RMS value of the image misfit between the DFT image and the 1000×1000 FFT dirty image made in `CASA` using the `clean` command with `iter`$= 0$. The orange line shows the RMS value of the image misfit when the FFT dirty image is the cropped `CASA` dirty image.

Figure 3.32 shows the RMS value of the image misfit versus the normalised map coordinate. The blue line shows the RMS value of the image misfit between the DFT image and the 1000×1000 FFT dirty image made in `CASA`. The orange line shows the RMS value of the image misfit when the FFT dirty image is the cropped `CASA` dirty image. The other three lines are identical to the lines with $W = 5, 6, 7$ in Figure 3.31 respectively.

Since the blue line in the figure reaches a misfit of 10^{-4} at around $x = 0.5$, compared to 10^{-2} achieved by spheroidal functions with $W = 5, 6, 7$ and $\alpha = 1$ when no cropping is performed, we conclude that the dirty image must have already been cropped in `CASA`. Further image cropping, which corresponds to the orange line, decreases the image misfit by much less than 10. It is not therefore necessary to perform extra image cropping on `CASA` dirty image products.

Offringa et al. (2014) stated that `CASA` uses the spheroidal function with a width of 7. For comparison, the image misfit RMS value using the spheroidal function with $W = 7$ (yellow line) is similar at 10^{-6} to that of `CASA` at the image centre. As the distance from the image centre increases, however, the `CASA` image misfit rises rapidly to 10^{-4} and remains at this level towards the image edge. We therefore believe that `CASA` uses a look–up table of spheroidal function samples. The rapid increase and the smaller image misfit indicates that the the spheroidal

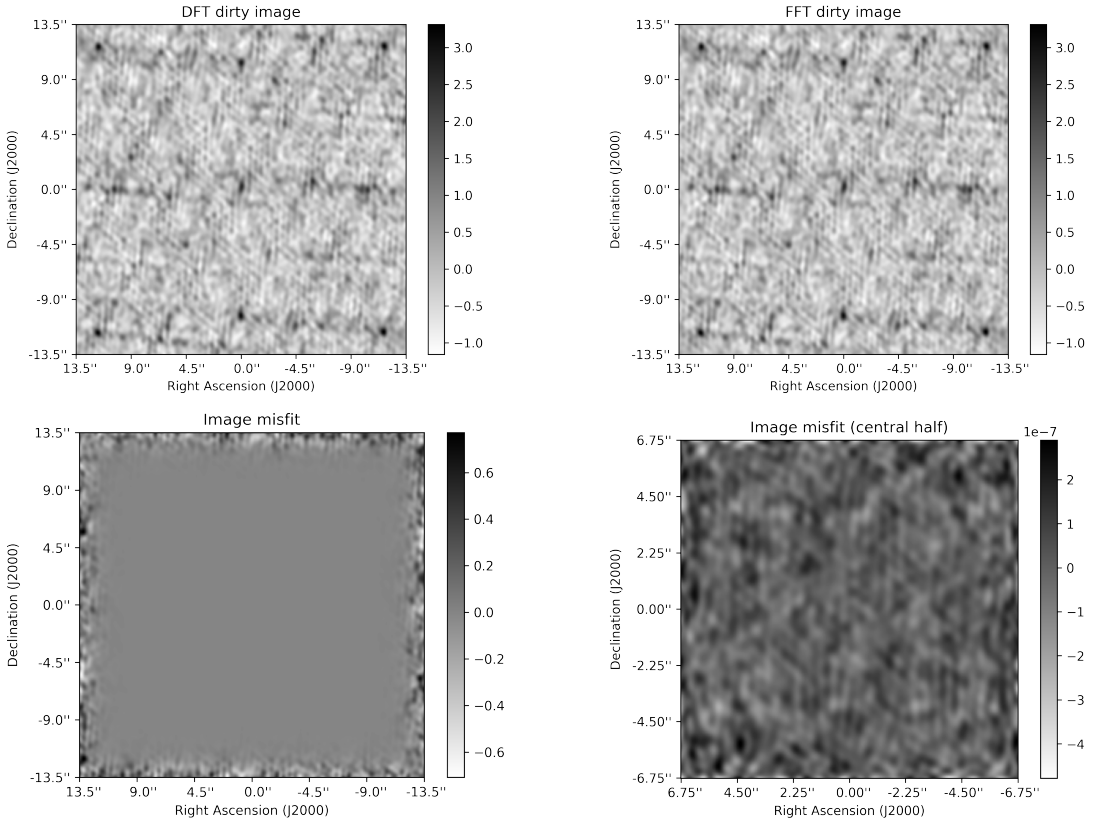


Figure 3.33: DFT dirty image (top left), FFT dirty image using the least–misfit gridding function with $W = 7$ (top right), and their difference (lower maps). The bottom right map is an enlarged version of the central half of the bottom left map. The dirty images were made from the simulated 34-source snapshot.

function is under–sampled. Further discussion of look–up tables and the choice of sampling rate is given in Section 3.5.

3.4.2.2 Image misfit comparison on simulated multiple–source data

We simulated more complex data so as to demonstrate the image misfits between the use of least–misfit gridding functions and spheroidal functions. The simulated data comprise a VLA snapshot observing 34 4 GHz point sources with variant fluxes and locations across the field of view. No noise sources or other corruptions were added. The simulation was also done in `CASA` with the same antenna configuration. More details of this dataset are given in Appendix C. The DFT dirty image of the data is shown in the top left figure in Figure 3.33, and the FFT dirty image using the least–misfit gridding function with $W = 7$ is shown at top right. The image misfit is shown at bottom left. The pixel size is 0.03 arcsec across the three 900×900 images. The bottom right map is an enlarged version of the central half of the image misfit map.

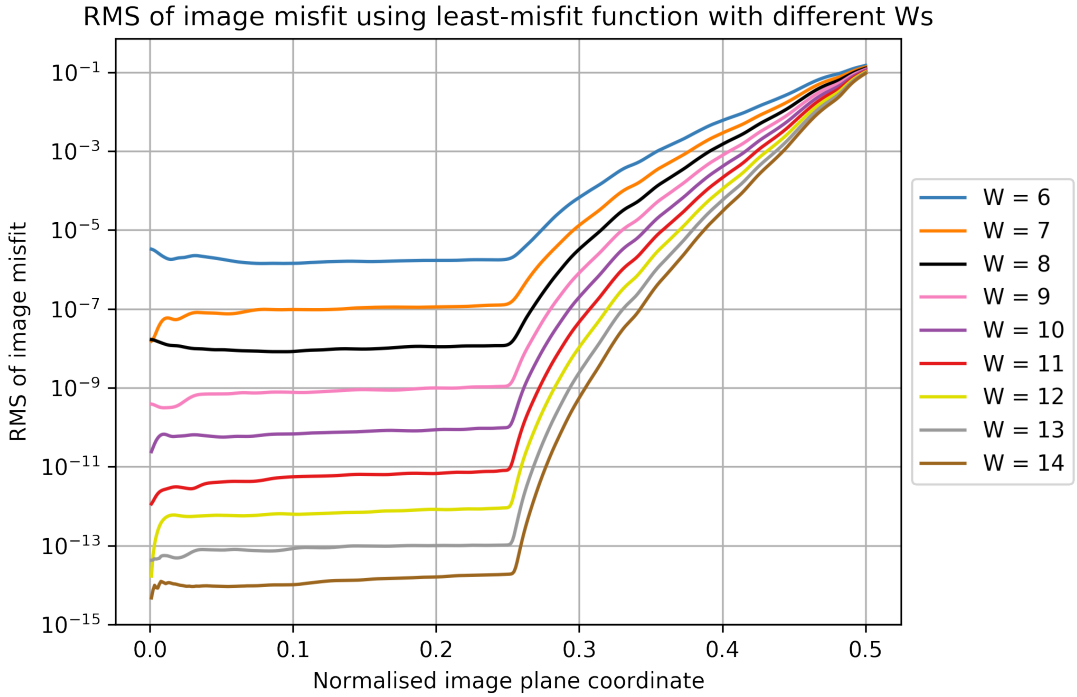


Figure 3.34: RMS value of the image misfit for 34 point–source simulated data. The gridding functions used are least–misfit functions with different values of W.

Figure 3.30 shows the RMS value of the image misfit for the simulated 34-source data. The gridding functions used are least–misfit functions with differing values of W. The x–axis represents the normalised image plane coordinate, and the y–axis is the RMS value of the image misfit. This figure shares the same features as Figure 3.30, but with larger image misfit because the dataset is more complex. Even so, the least–misfit function with $W = 7$ already reduces the image misfit to the limit of single precision in the range $[0, 0.25]$.

Figure 3.35 shows the RMS value of the image misfit for the same 34-source simulated data using spheroidal functions with different values of W. This figure also shares the same features as Figure 3.35, but with a slightly larger image misfit.

Upon comparing Figure 3.34 with Figure 3.35 in $[0, 0.25]$, we see that the image misfit using the least–misfit gridding function is at least 10^2 times better than using the spheroidal function with the same window width W.

We again construct two dirty images from the same data in `CASA`, one of which is twice as large as the other but is cropped so as to retain only the central half. We compare the `CASA` dirty images with the DFT image. Figure 3.36 shows the image misfit results. The results agree with our earlier conclusions on Figure 3.32.

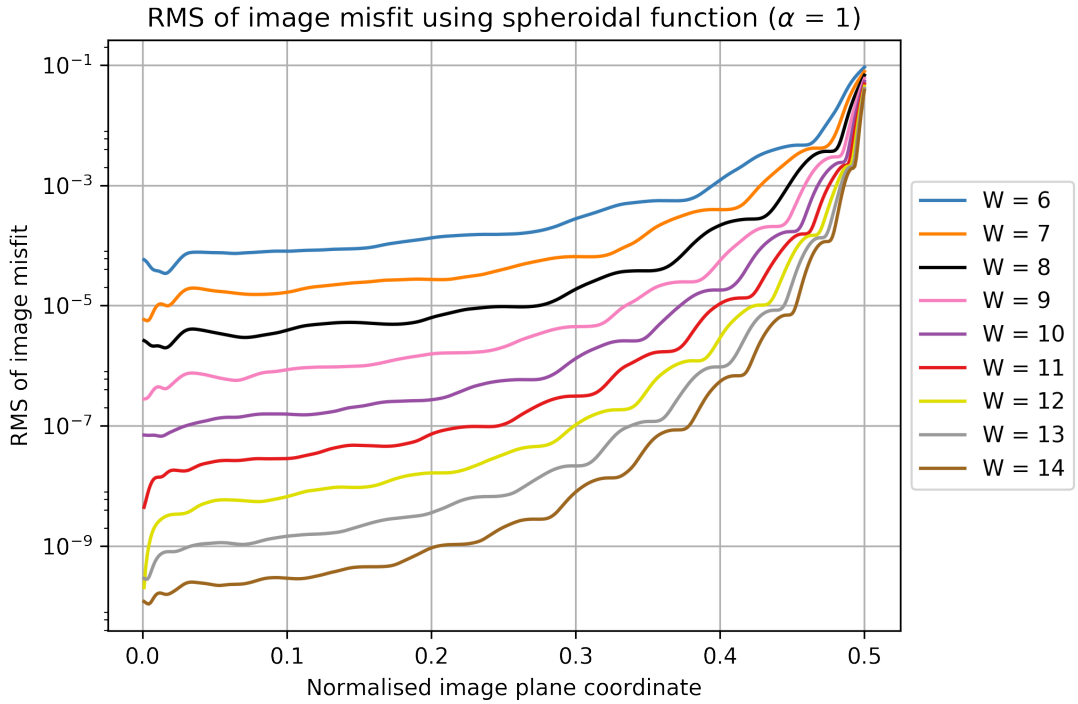


Figure 3.35: RMS value of the image misfit for 34 point–source simulated data. The gridding functions used are spheroidal functions ($\alpha = 1$) with different values of W.

3.4.2.3 Aliasing suppression: comparison of performance with simulated data

We discussed the theoretical aliasing suppression ratio in §3.4.1, and here we demonstrate the results of numerical experiments on aliasing suppression performance using simulated data. We consider a simple aliasing scenario in which there is no source within the field of view, and one point source outside it which causes aliasing within it. We simulate the data based on the VLA A–array configuration in `CASA` using function `simobserve`, identical to that used in Figure 3.1. There is only one point source, located at $(9.3'', 0'')$ with a flux of 2 Jy. Since the pixel size is 0.03 arcsec and the image size is 600×600, the source is 310 pixels distant from the image centre, therefore 10 pixels distant from the left image edge. No noise was added to this simulated data.

Figure 3.37 shows the aliasing caused by this point source lying outside the field of view. The top left figure shows the DFT dirty image of the simulated visibility data, in which the sidelobes of the outside source are very obvious. The top right figure shows the FFT dirty image; apart from the same sidelobes pattern, there is a bright source shown on the right, which is approximately 10 pixels horizontally from the right image edge. This is the alias of the 2–Jy point source 600 pixels away. The least–misfit function is used with $W = 7$ and $x_0 = 0.25$. The bottom left figure shows the difference between the DFT and FFT dirty images, where the alias

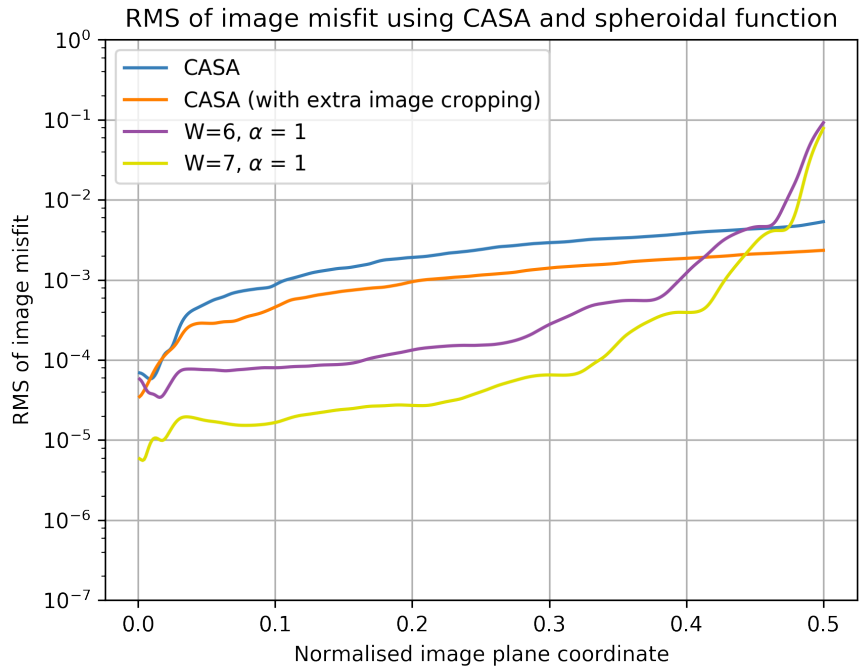


Figure 3.36: RMS value of the image misfit for the 34-source simulated data. The blue line shows the RMS value of the image misfit between the DFT image and the 900×900 FFT dirty image made in `CASA`. The orange line shows the RMS value of the image misfit when the FFT dirty image is the cropped `CASA` dirty image.

is very obvious with a maximum sky brightness of 0.82 Jy. The bottom right figure shows the central half of the image misfit, with a magnitude of approximately 10^{-8}.

If this point source is moved from very close to the edge to further away, we expect the brightness of the alias to decrease as a result of the aliasing suppression using the gridding function. We therefore made a set of point–source visibility data based on the same configuration, but with different positions of the source. The point source was kept outside the field of view, and its horizontal distance from the image edge was varied from 1 pixel to 298 pixels, but with its declination maintained as 0 arcsec. Sources located from 150 to 300 pixels away from the image edge would alias into the image range $[0, 0.25]$, and sources located from 0 to 150 pixels away would alias into the image range $(0.25, 0.5]$.

We constructed DFT and FFT dirty images from the datasets, and recorded the brightness of the aliases from the images of the differences. Figure 3.38 shows the normalised brightness of aliases versus the horizontal distance in units of pixel number from the source to the image edge. The least–misfit functions are used with different values of W.

The trend in the aliasing suppression shown in Figure 3.38 is consistent with the least–misfit gridding functions and map error functions in Figure 3.8. Aliases in the central half of the

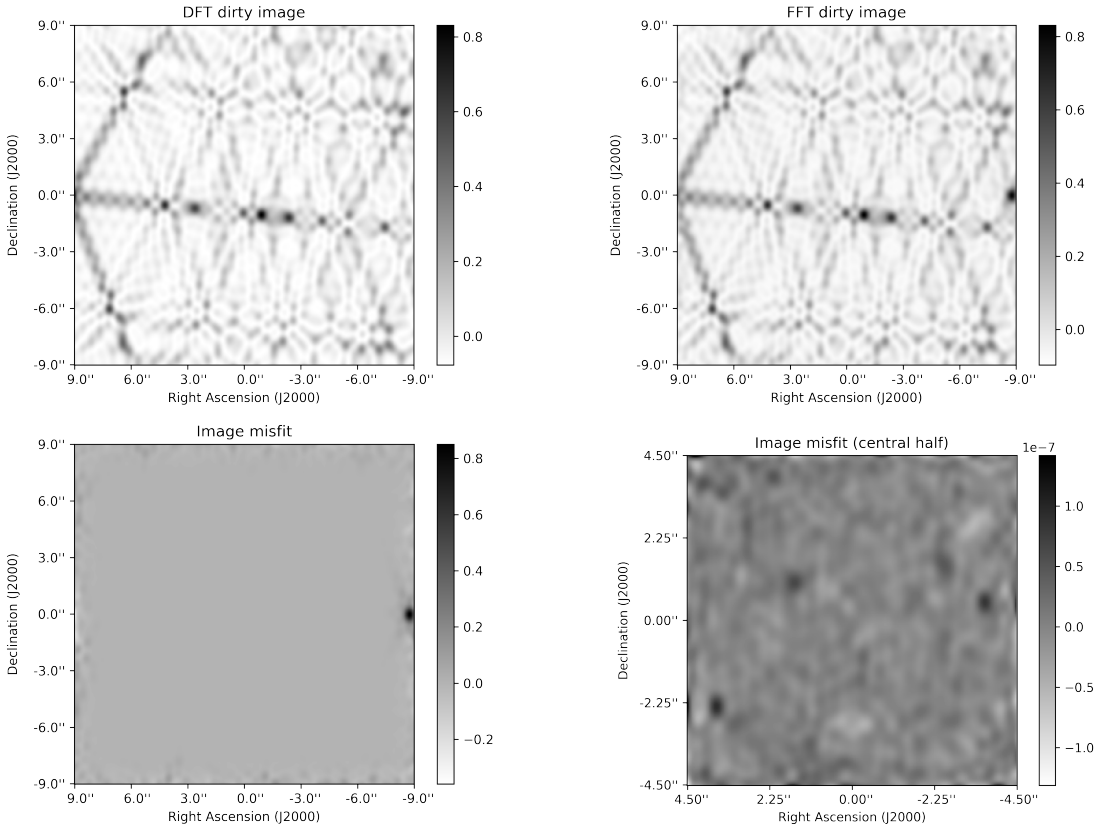


Figure 3.37: DFT dirty image (top left), FFT dirty image (top right) and the image misfit (bottom left) for the simulated visibility data. The alias is very obvious in the FFT dirty image with a maximum sky brightness of 0.82 Jy. The bottom right figure shows the central half of the image misfit with a magnitude of approximately 10^{-8}.

image are well suppressed, with a normalised brightness of the alias of approximately 10^{-7} for $W = 7$. The small fluctuations of the normalised brightness within $[0, 0.25]$ are also considered consistent with Figure 3.8.

As expected, the aliasing effect within the outer half of the image deteriorates as the distance between the source and the field edge decreases. Furthermore, as W increases, the least–misfit function becomes more powerful at suppressing aliasing. For $W \geq 7$, the alias brightness is already less than 10^{-7} of the original brightness within the range $[0, 0.25]$.

The same numerical experiment was then repeated using a set of different gridding functions. These were spheroidal functions ($\alpha = 1$) with different values of W. Figure 3.39 shows the results. It is consistent with the results in Figure 3.23: the image misfit in this case is the normalised aliasing brightness, and it grows with small fluctuations as the distance from the image centre increases. As W increases, the normalised aliasing brightness reduces. For $W \geq 11$, the alias brightness is a factor of at least 10^{-7} of the original brightness within the

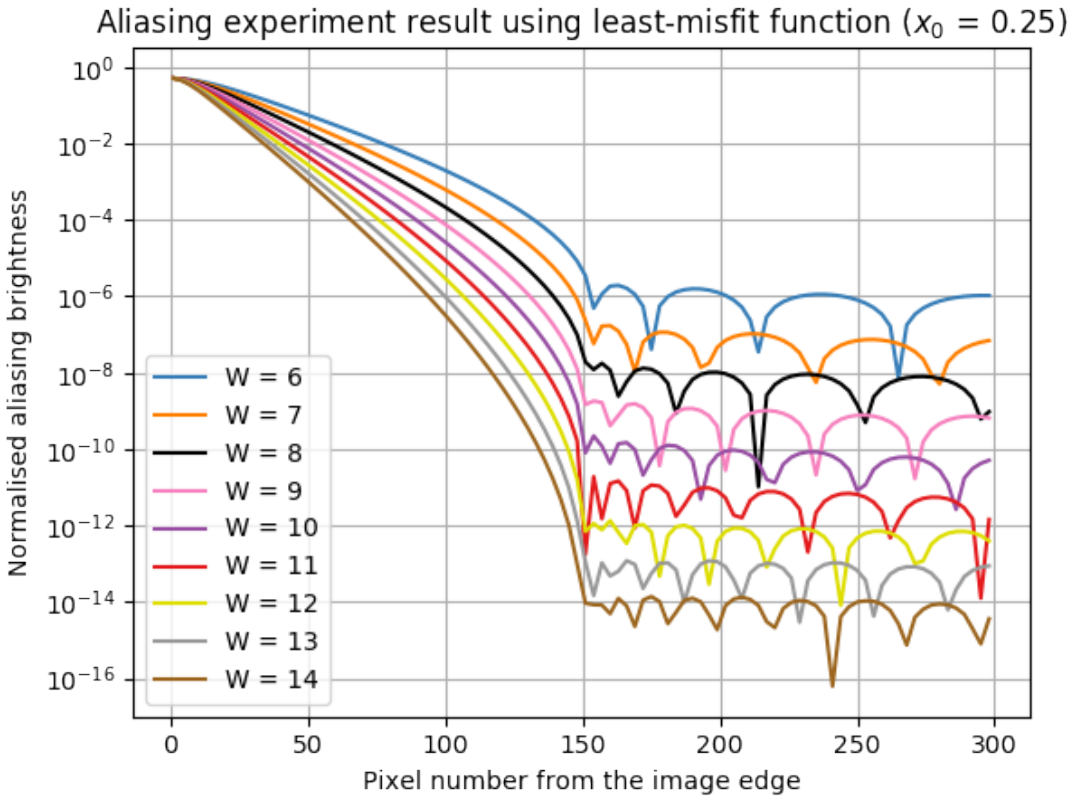


Figure 3.38: Normalised brightness of aliasing versus the horizontal distance from the point source to the edge of the field of view. Least–misfit functions with different values of W are used.

range $[0, 0.25]$.

By comparing the normalised aliasing brightness, we are able to compare quantitatively the aliasing performance of two different gridding functions. With W fixed, the least–misfit gridding function achieves at least 10^2 times smaller normalised aliasing brightness than the spheroidal function at the same position.

Upon combining this comparison with the numerical experiments on the aliasing suppression ratio in §3.4.1, we conclude that the least–misfit gridding function has better aliasing suppression performance than the spheroidal function with a same W.

3.5 Practical implementation of the least–misfit function

Gridding has long been viewed as the most computationally expensive procedure (Cornwell & van Diepen 2008) during the process of generating images in radio interferometry. The computational cost of the gridding process can be divided into two parts: the cost of calculating

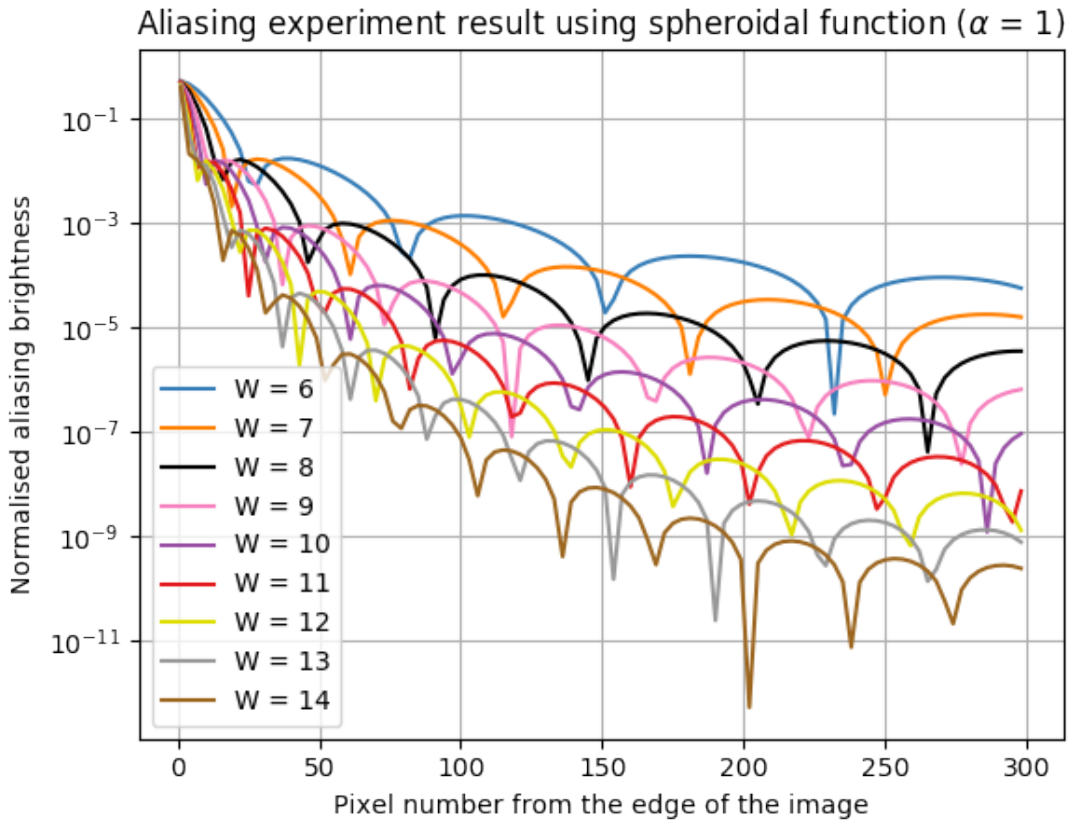


Figure 3.39: Normalised brightness of aliasing versus the horizontal distance from the point source to the edge of the field of view. The gridding functions used here are the spheroidal function ($\alpha = 1$) with different values of W. The x–axis represents the number of pixels from the point source to the image edge, and the y–axis is the normalised aliasing brightness.

gridding functions, and the cost of the gridding process, which is significantly greater.

In practice, the gridding function can be pre–calculated and saved in a look–up table. For example, a gridding function $C(u)$ with a support width $W = 8$ can be uniformly sampled at $8M_s$ u values, with M_s being the sampling number for each grid. We refer M_s as the 'sampling rate'. In view of the symmetry of $C(u)$, it is necessary to store only half of these values. This has to be done only once, and the result can be retrieved for use with different observation data.

Given the visibility number N_v, image pixel size N_x, N_y and the width of the gridding function W, the gridding process using a gridding function look–up table C can be written in pseudocode as follows:

```
For k from 1 to N_v:
    Find indices u_index and v_index for the kth visibility;
    Find indices for W adjacent grid points on both the u and v axis;
    Take the corresponding gridding values from C using the indices;
```

```
5   For m from 1 to W:
6     For n from 1 to W:
7       Compute gridded data via convolutions on both the u and v axes;
8     End
9   End
10  Sum the existing value and gridded value at each grid;
11 End
```

In practice, the process can be separated into a gridding setup (Lines 1 to 4), a convolution process (Lines 5 to 7) and a sum (Line 8). We now explain the gridding setup steps. Since the look–up table has already been stored, we need indices u_{index} and v_{index} of the look–up table to locate the u and v value for each item of visibility data via nearest–neighbour search; see Line 2. Next, the indices are specified for W adjacent grid points on the u–axis and W adjacent grid points on the v–axis, onto which the given visibility data will be gridded (Line 3). The corresponding $2W$ gridding function values will be obtained, see Line 4. At the end of the process, the gridded N_x by N_y visibilities and a table containing $2WN_v$ gridding function values are stored. This table is referred to as the 'setup table'.

As for the sum (Line 8), it is usually considered as a part of the convolution process. After assigning a specific visibility value onto the W^2 grid points, the existing value at these grid points will be summed up with the new assigned gridded values. The inverse FFT will be applied to these summed values at Cartesian grids.

In this section, we first discuss the choice of the sampling rate M_s. Then we discuss the computational cost of the convolution process. After that, the FFT computational cost will be discussed briefly. As a result of considering the computational costs of the convolution and FFT, we recommend use of the least–misfit gridding function with $W = 7$ and $x_0 = 0.25$ to achieve an image misfit at single floating point precision. To achieve the same precision using spheroidal functions would require considerably more computation and storage costs.

3.5.1 Look–up table and sampling rate

The least–misfit gridding function is calculated through an optimisation process, so that a look–up table saves the computational time of the optimisation. We use the simplest nearest neighbour method to retrieve data from the look–up table. Compared to other interpolation methods such as linear, quadratic or cubic interpolation, this requires less memory access and lower calculation cost. It has one obvious disadvantage, however: a fairly large sampling rate M_s is necessary to reach the desired accuracy.

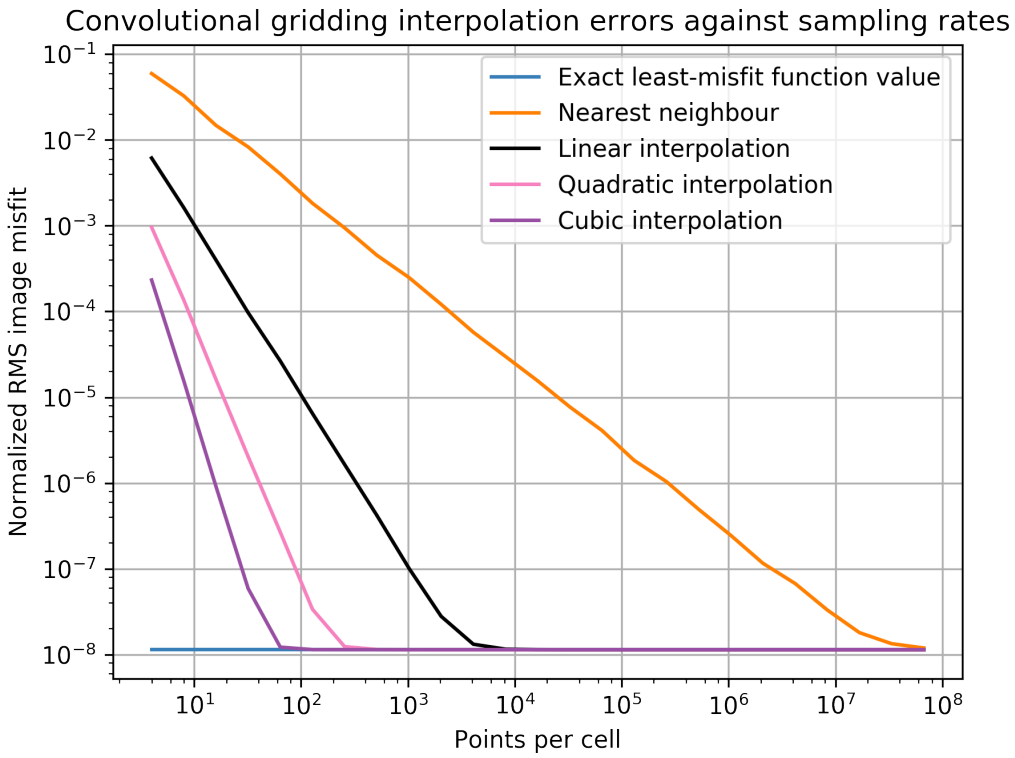


Figure 3.40: The sampling rate of the gridding function look–up table impacts the accuracy of the FFT dirty image, compared to the DFT dirty image. The gridding function used is the least–misfit gridding function with $W = 8$ and $x_0 = 0.25$. Four different interpolation methods are used. The result using exact gridding function values is also shown as a solid blue line in the plot, as a benchmark.

A numerical experiment was designed to explore the relation between the RMS value of the image misfit and the sampling rate, using different interpolation methods. The data used comprise 1000 randomly distributed complex visibilities over a single–dimension u, where the values of u are also generated from a random distribution. Four different interpolation methods are implemented with different sampling rates. Their RMS values for the image misfit will be compared against the result using the exact least–misfit gridding function values ($W = 8, x_0 = 0.25$).

Figure 3.40 shows the results. The x–axis is the number of points per cell, or M_s, and the y–axis is the normalised RMS image misfit within the range $[0, x_0]$. For all four interpolation methods, if the sampling rate M_s is too small then the image misfit is unable to match the results using the exact gridding function values. If nearest neighbour method is used, the sampling rate M_s should be around 10^6 in order to reach the single precision limit. For the linear, quadratic and cubic interpolations, the sampling rates needed are much smaller.

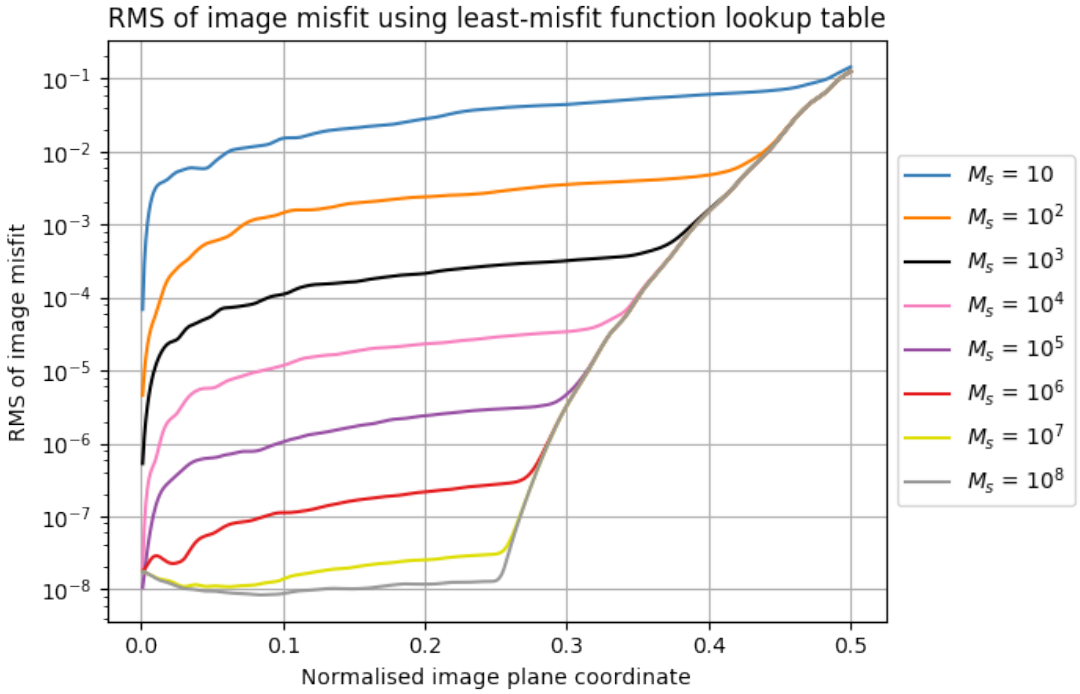


Figure 3.41: RMS value of the image misfit achieved using look-up tables of the least–misfit function ($W = 8, x_0 = 0.25$) with different sampling rates. For $M_s = 10^6$, the stored least–misfit function already makes the image misfit to achieve the single precision limit.

Although a sampling rate of 100 already reduces the normalised RMS value of the image misfit to 10^{-8} by cubic interpolation, the extra computational cost and memory access involved in cubic interpolation results in a longer gridding setup time, especially for large datasets. In contrast, no extra cost is incurred by nearest neighbour method other than storage cost. Although the storage space of its look–up table is larger, this is highly acceptable: 10^6 single precision numbers cost less than $5M$ storage. Since a sampling rate of 10^6 allows the normalised RMS value of the image misfit to reach the single precision limit, we recommend the use of a larger look–up table via nearest neighbour method.

A more complicated two–dimensional simulated data is then tested using the least–misfit gridding function ($W = 8, x_0 = 0.25$) with look–up tables. This simulated data is identical to the 34-source VLA data used in Section 3.4.2. We change the sampling rate M_s to determine the corresponding accuracy levels that our tabled gridding functions are able to achieve. Figure 3.41 shows the image misfit RMS value for different sampling rates. When $M_s = 10^6$, the tabled least–misfit function already allows the image misfit to reach the single precision limit. When $M_s = 10^8$, the same accuracy is attained as by the exact gridding function; see Figure 3.34.

In Figures 3.42 and 3.43, we plot the RMS values for the image misfit using look–up tables

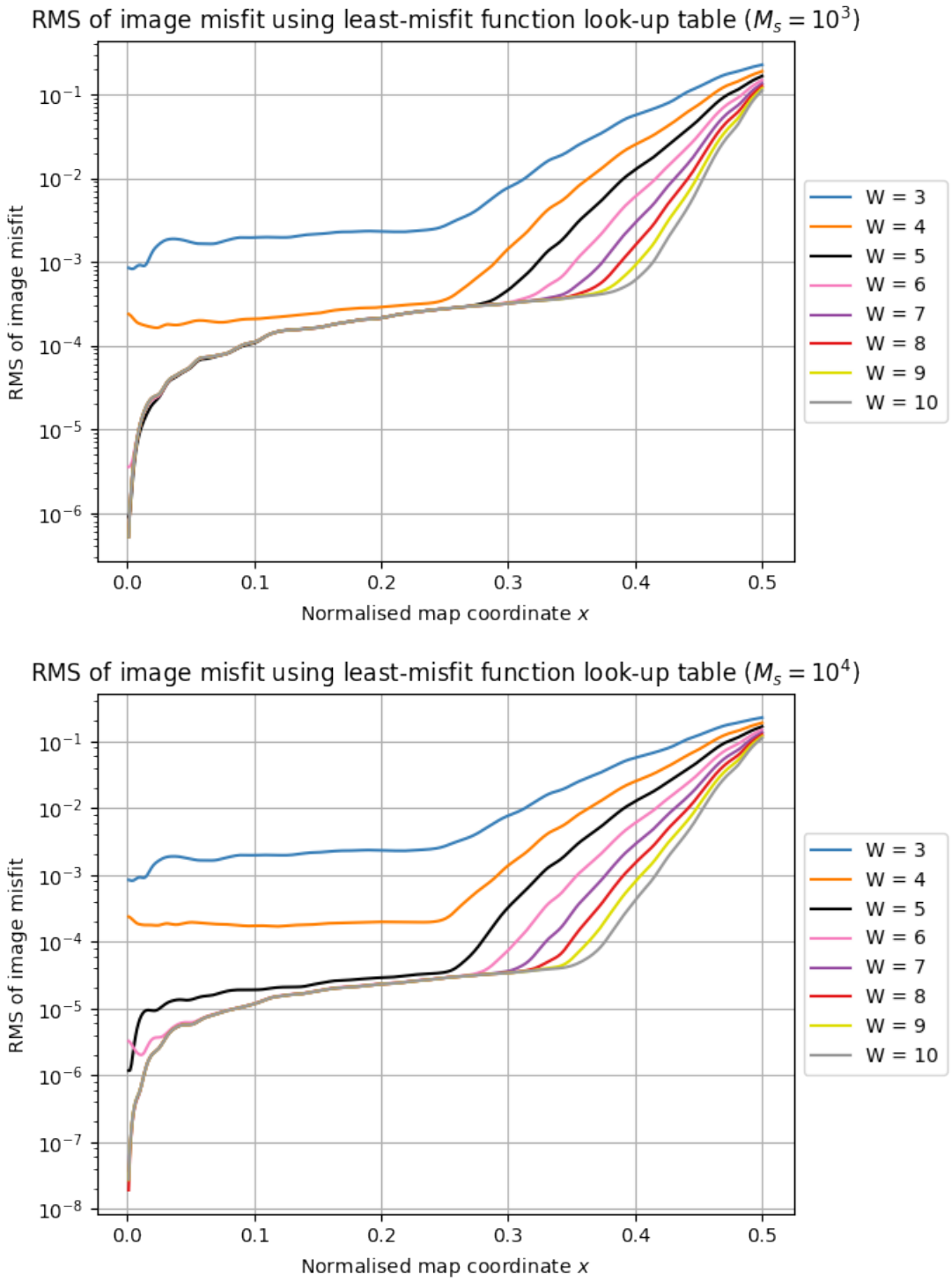


Figure 3.42: RMS of the image misfit achieved using look–up tables of the least–misfit function with differing support width W. The sampling rates for the upper and lower plots are respectively $M_s = 10^3$ and $M_s = 10^4$.

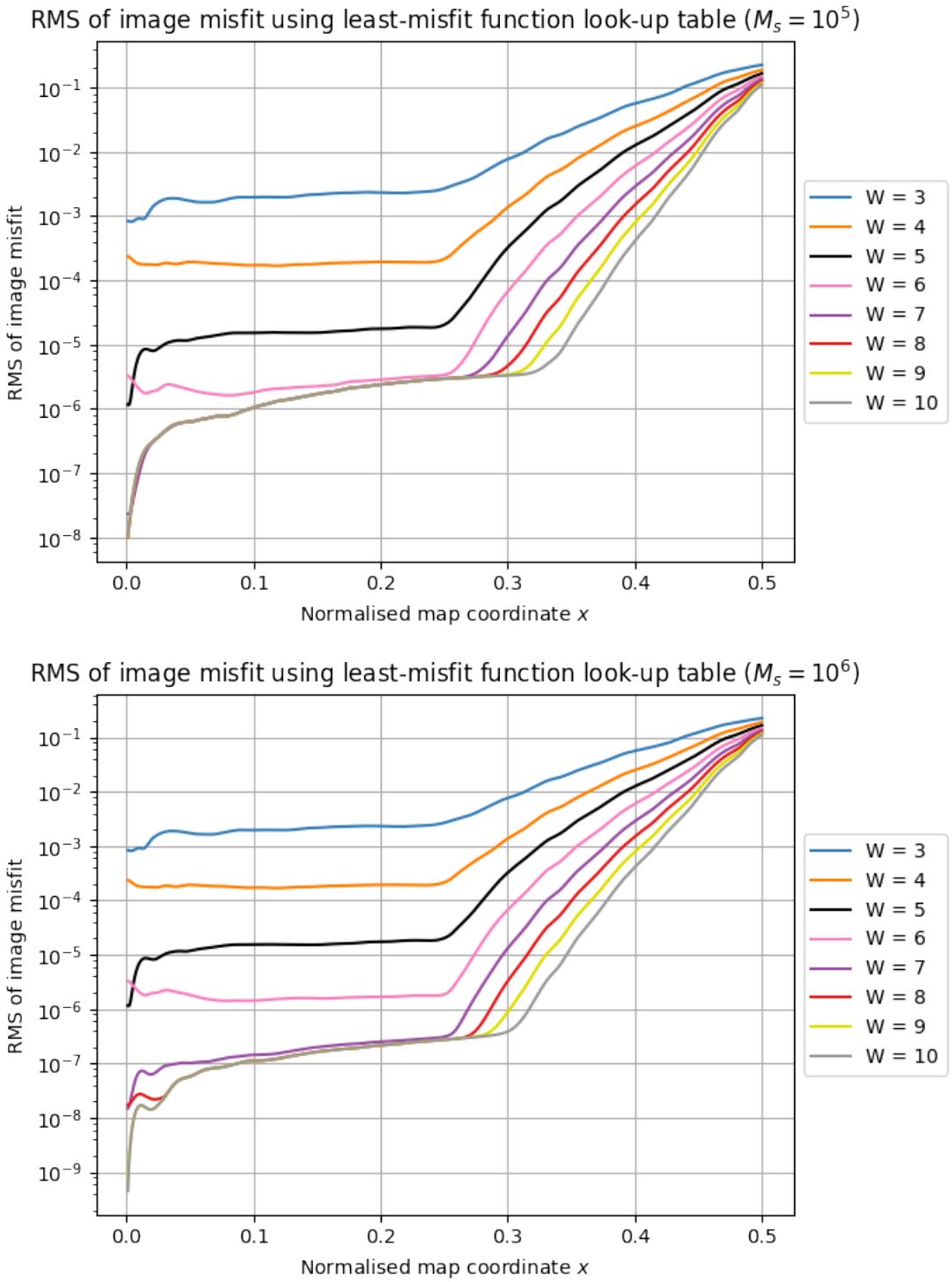


Figure 3.43: RMS of the image misfit achieved using look–up tables of the least–misfit function with differing support width W. The sampling rates for the upper and lower plots are respectively $M_s = 10^5$ and $M_s = 10^6$.

of the least–misfit function with the same sampling rate M_s but different values of the support width W. Larger values of W produce more accurate results when exact gridding function values are used, as seen in Figure 3.34, but when the sampling rate is as small as 10^3, $W > 4$ can achieve an accuracy of only 10^{-4} at most. In conclusion, the small sampling rate stops the gridding function with a larger W to achieve its best performance. If the goal is to only attain such accuracy, a $W = 4$ least–misfit gridding function is already satisfactory with a sampling rate $M_s = 10^3$.

With the increase of the sampling rate, we can see that the gridding function with a larger W achieves greater accuracy, and we recommend using $W = 7$ and $M_s = 10^6$ instead of $W = 8$; a larger W leads to extra computational cost, as explained in the next section.

In Figure 3.44, we demonstrate a FFT dirty image using the least–misfit gridding function with $W = 8$ and $x_0 = 0.25$ in the top left figure. A look–up table is used with a sampling rate of only 100. The data used is the same 34–source simulated VLA A-array data as that used in Figure 3.33. The difference between the DFT dirty image and FFT dirty image is displayed in the top right figure, and its central one third is enlarged and displayed in the lower left image. It is noted that the RMS image misfit deteriorates from the centre of the map, as can be seen in the enlarged figure. Let us return to the image misfit shown in the right bottom map in Figure 3.33, when no look–up table is in use, the image misfit has a magnitude of $10^{-7ki\cdot}$ all across the central half of the full image. This significant difference comes from the use of look–up table.

Also, we can see in Figure 3.41, the RMS image misfit builds up away from the centre of the map. It is especially obvious closer to the image centre. For example, with M_s fixed, the RMS of the image misfit increases rapidly from $x = 0$ to $x = 0.05$, then sees a more steady and mild rise to $x = x_0$.

Here we analyse the accuracy of the dirty beam to explain this effect. A sampled convolution function $C(u)$ is used. It is not necessary to consider the accuracy of the dirty image separately, as dirty image can be considered as a weighted dirty beam, with weights being the visibilities.

The essence of the effect can be seen by calculating what effect an offset $u_k \rightarrow u_k + \delta u_k$ has on the dirty beam. The one–dimensional dirty beam can be written as $B(x) = \sum_{k=1}^{M} w_k \cos(2\pi x u_k)$, where w_k is the weight for baseline k. Using the small angle approximation, we obtain

$$B(x) + \delta B(x) = \sum_{k=1}^{M} w_k \cos[2\pi x(u_k + \delta u_k)] \approx \sum_{k=1}^{M} w_k [\cos(2\pi x u_k) - 2\pi x \delta u_k \sin(2\pi x u_k)] \tag{3.28}$$

The difference $\delta B(x)$ clearly has zero mean. Assuming all the $\langle \delta u_k^2 \rangle = \delta u^2$ are equal and

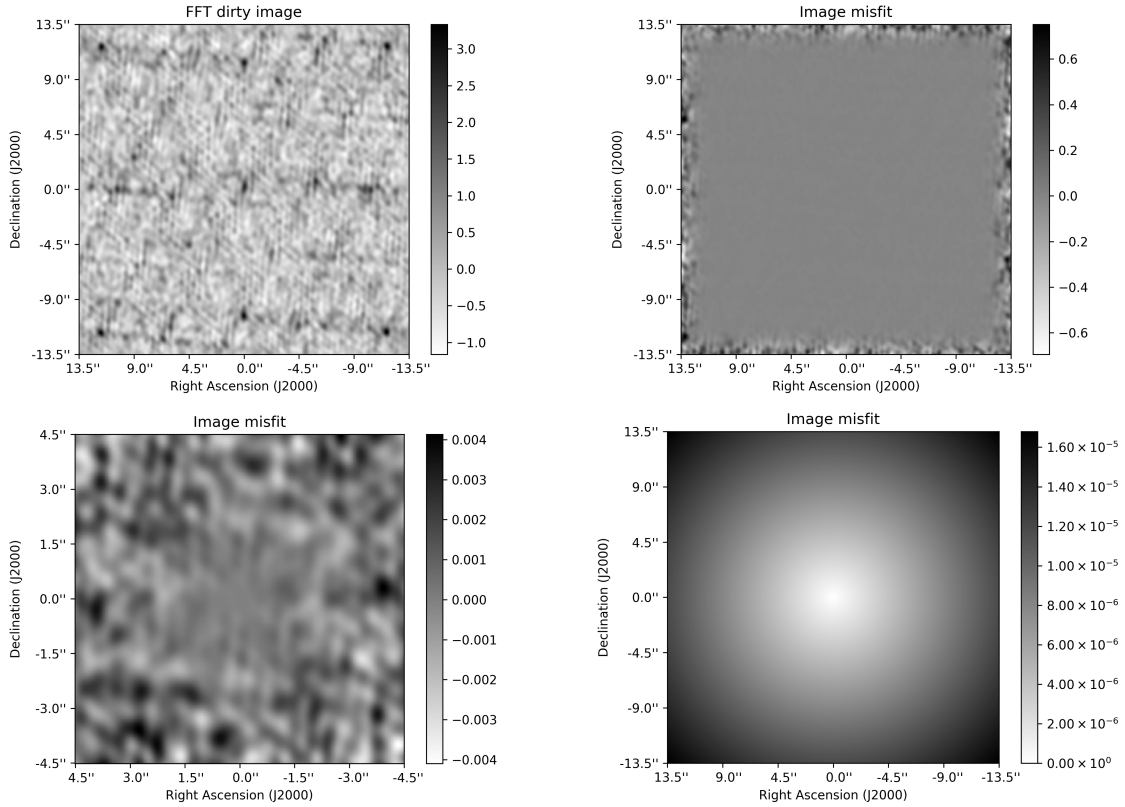


Figure 3.44: FFT dirty image using the least–misfit gridding function with $W = 8$ and $x_0 = 0.25$ is demonstrated in the top left figure. A look–up table is used with a sampling rate of only 100. The difference between the DFT dirty image and FFT dirty image is displayed in the top right figure, and its central one third is enlarged and displayed in the lower left image. The bottom right map shown is $\sqrt{\langle \delta B(x,y)^2 \rangle}$ according to Equation 3.30.

uncorrelated between samples k, the variance can write

$$\langle \delta B(x)^2 \rangle = 4\pi^2 x^2 \delta u^2 \sum_{k=1}^{M} w_k^2 \sin^2(2\pi x u_k) \tag{3.29}$$

Generalising it to 2 dimensions, and assuming δu is not correlated with δv we find

$$\langle \delta B(x,y)^2 \rangle = 4\pi^2 (x^2 \delta u^2 + y^2 \delta v^2) \sum_{k=1}^{M} w_k^2 \sin^2[2\pi(x u_k + y v_k)] \tag{3.30}$$

Measured in pixels, we set $\delta u = \delta v = 1/(\sqrt{12} M_s)$ due to the level of quantisation noise. We assign the weights to be 1 and $M_s = 100$. According to Equation 3.30, we demonstrate the $\sqrt{\langle \delta B(x,y)^2 \rangle}$ in the bottom right map in Figure 3.44. The pattern shown on the map explains the effect well: the variance of the beam builds up from the centre towards the edge, resulting in better accuracy at the image centre.

Return to Equation 3.29, we can find that the logarithm of the beam variance is linearly dependent on $-\log(M_s)$ given that δu is limited by $1/(\sqrt{12} M_s)$. This linear relationship is

confirmed by Figure 3.40, where the orange line representing the nearest–neighbour method has a slope of -1. The figure also shows clearly that the logarithm of the RMS image misfit is linearly dependent on $-2\log(M_s)$ for linear interpolation, and $-3\log(M_s)$ for quadratic interpolation and so on. An alternative derivation is given as follows to explain these relationships.

The one–dimensional FFT dirty beam can be written as

$$B(x) = h(x)\sum_k^M w_k \sum_{n\in S_k} C(n-\nu_k)\cos(2\pi n x) = h(x)\sum_k^M w_k \sum_r C_r(\nu_k)\cos[2\pi(r-\nu_k)x] \tag{3.31}$$

where $C_r(\nu) \equiv C(r-\nu)$.

When the nearest–neighbour method is in use, with Taylor expansion we have $C_r(\nu_k+\delta\nu_k) \approx C_r(\nu_k) + \delta\nu_k C_r'(\nu_k)$. The difference δB becomes:

$$\delta B(x) = h(x)\sum_k^M w_k \sum_r C_r'(\nu_k)\delta\nu_k \cos[2\pi(r-\nu_k)x] \tag{3.32}$$

Assuming all the $\langle\delta\nu_k^2\rangle = \delta\nu^2$ are equal and uncorrelated between samples k, the variance can write

$$\langle\delta B(x)^2\rangle = \delta\nu^2 h(x)^2 \sum_k^M w_k^2 \sum_r \sum_s C_r'(\nu_k)C_s'(\nu_k)\cos[2\pi(r-\nu_k)x]\cos[2\pi(s-\nu_k)x] \tag{3.33}$$

This proves the linear relationship in logarithm between the beam variance and the sampling rate, given $\delta\nu$ is also limited by $1/(\sqrt{12}M_s)$.

When the linear interpolation is in use, with Taylor expansion we have $C_r(\nu_k+\delta\nu_k) \approx C_r(\nu_k) + \frac{1}{2}\delta\nu_k^2 C_r''(\nu_k)$. Proceeding as above, we find the difference as

$$\delta B(x) = \frac{1}{2}h(x)\sum_k^M w_k \sum_r C_r''(\nu_k)\delta\nu_k^2 \cos[2\pi(r-\nu_k)x] \tag{3.34}$$

, and the variance as

$$\langle\delta B(x)^2\rangle = \frac{1}{4}\delta\nu^4 h(x)^2 \sum_k^M w_k^2 \sum_r \sum_s C_r''(\nu_k)C_s''(\nu_k)\cos[2\pi(r-\nu_k)x]\cos[2\pi(s-\nu_k)x] \tag{3.35}$$

We can find that the logarithm of the beam variance is linearly dependent on $-2\log(M_s)$ given that $\delta\nu$ is limited by $1/(\sqrt{12}M_s)$. This linear relationship is consistent with Figure 3.40, where the black line representing the linear interpolation has a slope of -2. The quadratic interpolation with a slope of -3 can also be explained in this way.

Equation 3.24 is used to evaluate $C_r(\nu)$ here, so we have its first derivative as:

$$C_r'(\nu) = 2\pi\sum_{r'} A_{rr'}^{-1}\int_{-x_0}^{x_0} \mathrm{d}x h(x) x \sin[2\pi(r'-\nu)x] \tag{3.36}$$

and the second derivative as

$$C_r''(v) = 4\pi^2 \sum_{r'} A^{-1}_{rr'} \int_{-x_0}^{x_0} \mathrm{d}x h(x) x^2 \cos[2\pi(r' - v)x] \tag{3.37}$$

We find discontinuities of $C_r'(v)$ and $C_r''(v)$ at the cell boundaries visible to much higher W than in $C_r(v)$.

The linear interpolation is worth considering, because it brings much less extra computational cost and memory access than the cubic interpolation, and only requests a sampling rate of 10^3 to achieve the single precision. We will focus on the nearest–neighbour method for now. The discussion of the linear interpolation will be presented in future papers that are in draft at the time of writing.

3.5.2 Computational cost of the convolution process

We now calculate the computational cost of the gridding process, and then the storage cost. Following Yashar & Kemball (2009), the computational cost depends on the visibility output rate, the intensity of computing operations and the efficiency of parallel computing. We focus only on the computing intensity here and do not consider the data rate or parallel computing[d]. In the present work, the computational cost of a process refers to how many operations the process involves.

The number of operations for the gridding setup is $4WN_v$, including $2WN_v$ operations to find indices and $2WN_v$ operations to retrieve gridding function values. The convolution process requires W^2N_v operations along with W^2N_v sums. The storage cost is simple: the setup table contains $2WN_v$ gridding function values, and there are the gridded $N_x \times N_y$ visibilities. The cost is summarised in Table 3.9. We assign 8 bytes to each visibility (because of the real and imaginary part), and 4 bytes to each gridding function value.

Table 3.9: Computational and storage cost of the gridding process

Process	Operations	Storage cost (byte)
Gridding setup	$4WN_v$	$8WN_v$
Convolution	$2W^2N_v$	$8N_xN_y$

We tested the 34-source simulated data on a computer with the quad–core Intel i5-2310 @ 2.90GHz processor and 8 GB of memory. If $M_s = 10^6$, it takes only 13 seconds to build the set–up table, and 20 ns for each visibility to be gridded. On a computer with a 6-core Intel i7-4930 processor and 12GB of memory, the equivalent gridding time is reduced to 15.4ns

[d]Discussions of parallel computing during the gridding process are given in Romein (2012); Varbanescu et al. (2018)

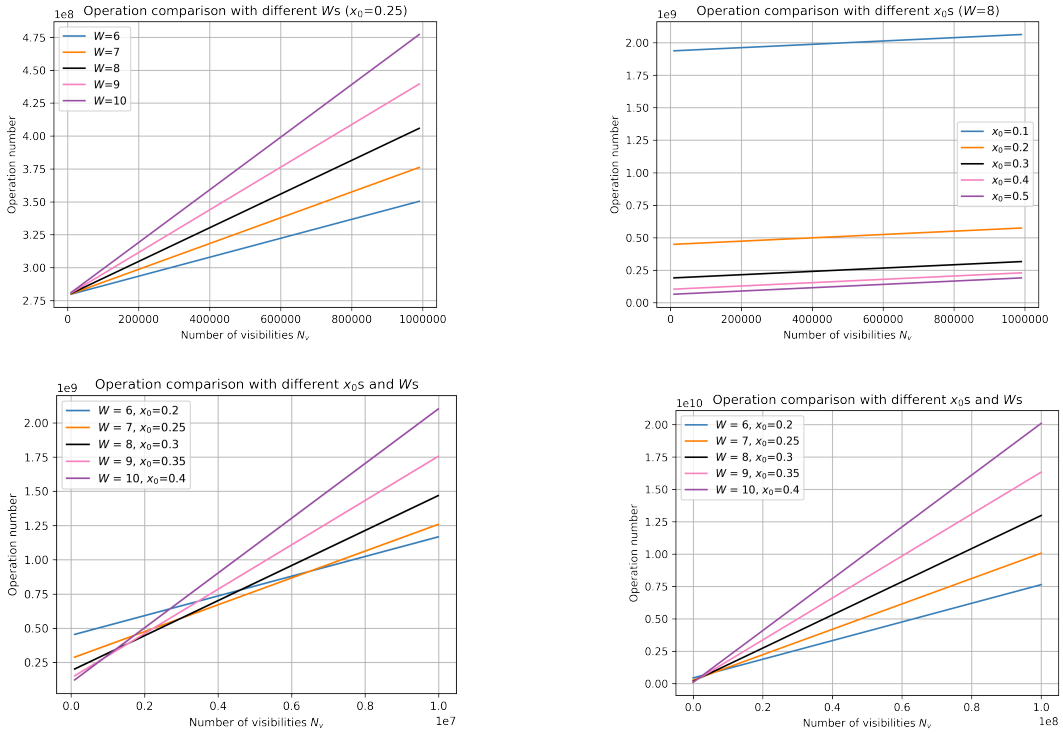


Figure 3.45: Computational and storage cost of the gridding process

per visibility. For any given observation, multiple convolutions may be needed for different imaging purposes, but the gridding setup is usually needed only once. We consider only the convolution cost.

We next consider the FFT cost. Since the FFT image will be cropped from $x = x_0$ to $x = 0.5$ to make a final image of size N_x by N_y, it is necessary to make a $N_x/(2x_0)$ by $N_y/(2y_0)$ FFT dirty image[e]. Since $x_0, y_0 \in [0, 0.5]$, it follows that for smaller x_0 and y_0 the FFT computational cost is larger. We tested the same 34-source simulated data on the computer with 6-core Intel i7-4930 processor and 12GB memory, which took $20ns \times N_x N_y N_v$ to compute the DFT dirty image, effectively taking $150ns$ to compute the FFT value on each pixel with $x_0 = 0.25$.

For greater accuracy in the image misfit, W should be larger and and x_0 smaller. Larger W results in greater computational cost of the convolution, however, and a smaller x_0 incurs higher FFT computational costs. A compromise is necessary to achieve acceptable image misfit at an affordable computational cost.

To achieve single precision accuracy, it is not necessary to set $W = 7$ and $x_0 = 0.25$. In fact we can achieve the desired precision if we reduce W to 6 and x_0 to 0.2. The smaller x_0 increases the FFT computational cost, and the decrease in W results in lower convolution

[e] y_0 is usually equal to x_0.

cost. Which effect is more significant? We combine the convolution and FFT operations as $N_v(W^2) + \frac{N_x N_y}{4x_0 y_0} \log(\frac{N_x N_y}{4x_0 y_0})$. It is assumed that $x_0 = y_0$.

We first fix $x_0 = 0.25$ and change W so as to plot the computational cost in the top left plot of Figure 3.45. The image size is set at $N_x = N_y = 2048$, and the visibility number is changed from 10^3 to 10^6. We observe that W defines the slope of the linear relation between the operations and the visibility number. For larger W the gradient is larger, and the computational cost is therefore greater.

We then fix $W = 8$ and vary x_0, in the top right plot. The image size and visibility number are unchanged. As W is fixed, the gradients of the five lines are the same, but the y–intercepts differ depending on the value of x_0. Larger values of x_0 imply less image cropping, so that the computational cost is smaller.

Next, we choose 5 sets of W and x_0 pairs, by which the image misfit reaches the single precision limit. Their computing cost is plotted as numbers of operations in the bottom two plots in Figure 3.45, with visibility number ranging from 10^4 to 10^7 at left and 10^4 to 10^8 to the right. The bottom right plot shows that W dominates the gridding computational cost when the visibility number is greater than 10^7. We see $W = 6$ and $x_0 = 0.2$ plotted in blue in the bottom right plot; although the corresponding FFT image has to be cropped more than any other W and x_0 pair, its computational cost is the smallest because W is smallest.

In the bottom left plot, when N_v is small, x_0 dominates the computational cost. Then, as N_v increases, W comes to dominate the cost. The computational cost of using $W = 8, x_0 = 0.3$ and $W = 7, x_0 = 0.25$ remain relatively small within the given range of visibility numbers. We prefer to use $W = 7, x_0 = 0.25$, for a smaller look–up table and setup table.

Based on these results, we recommend using $W = 7$ and $x_0 = 0.25$ to achieve the single floating point precision limit in the image misfit between the DFT and FFT dirty images. This recommendation is based on good balance between the image accuracy and the computational cost.

In comparing the least–misfit function against the spheroidal function, it is clear that if x_0 is fixed, we are forced to choose a much larger W for the spheroidal function in order to achieve the same level of image misfit, thereby incurring extra computational and storage cost. If we choose the same W for both gridding functions, better precision is obtained by using the least–misfit function. With the same computational cost, therefore the least–misfit function is preferred.

Table 3.10: RMS value of the degridding misfit when using the least–misfit gridding function with different values of W (double precision)

W	RMS of the visibility misfit
1	4.16×10^{-1}
2	4.40×10^{-2}
3	2.91×10^{-3}
4	2.62×10^{-4}
5	2.76×10^{-5}
6	1.85×10^{-6}
7	1.76×10^{-7}
8	1.70×10^{-8}
9	1.83×10^{-9}
10	1.29×10^{-10}
11	8.69×10^{-12}
12	1.49×10^{-12}
13	1.64×10^{-13}
14	1.46×10^{-14}

Table 3.11: RMS value of the degridding misfit when using the least–misfit gridding function with different values of W (single precision)

W	RMS of the visibility misfit
1	4.16×10^{-1}
2	4.40×10^{-2}
3	2.92×10^{-3}
4	2.70×10^{-4}
5	2.85×10^{-5}
6	1.84×10^{-6}
7	2.98×10^{-7}
8	2.28×10^{-7}

3.6 Degridding with the least–misfit gridding function

The degridding process is the mathematical transpose of the gridding process. When using the least–misfit function with $x_0 = 0.25$, the corresponding degridding steps can be summarised as:

1) Multiply the image model with the correcting function.
2) Pad zeros to the outer edge of the corrected model image to make it double–sized.
3) Apply the FFT to the corrected image.
4) Correlate the FFT results with the gridding function to obtain the visibility model.

In this section, we apply our least–misfit gridding function to the degridding process. We first demonstrate the degridding performance using the least–misfit gridding function, and then

Table 3.12: RMS value of the visibility misfit using the least–misfit gridding ($W = 7$, $x_0 = 0.25$) look–up table with different sampling rates M_s

M_s	RMS of the visibility misfit
10^2	6.23×10^{-3}
10^3	4.48×10^{-4}
10^4	4.40×10^{-5}
10^5	4.62×10^{-6}
10^6	4.85×10^{-7}
No sampling	2.98×10^{-7}

discuss the effect of different sampling rates. Comparison will also be made with the spheroidal function.

The 'degridding misfit', or 'visibility misfit', is defined as the difference between the original visibilities and the degridded results from the given image model. We take the RMS value of the misfit; a smaller RMS value of the degridding misfit indicates better degridding performance. In the following degridding experiment, we reuse the VLA A–array 34 point–source simulated data. The corresponding image model is a simple 450 by 450 matrix in which all components are zero except for 34 values assigned to where the sources are, representing their fluxes. In the experiment, the image model is multiplied by the correcting function $h(x)h(y)$ and then gets zero–padded to 900×900; this is followed by an FFT and a correlation to give the degridded visibility.

If double precision is used, the RMS values of the visibility misfit using the least–misfit gridding function with different values of W are listed in Table 3.10. The results when single precision is used are set out in Table 3.11. We see that use of the least–misfit gridding function with $W = 7$ and $x_0 = 0.25$ in single precision already causes the visibility misfit to reach the single precision limit. The choice of $W = 8$ improves the visibility misfit slightly, but at greater computational cost.

When a look–up table is used instead of the exact gridding function values, the RMS values of the visibility misfit using the least–misfit gridding function (with $W = 7$, $x_0 = 0.25$) are shown in Table 3.12. We see that a sampling rate of 10^6 already causes the visibility misfit to achieve the single precision limit, which is consistent with our recommendation during the gridding process.

In summary, we recommend using $W = 7$ and $x_0 = 0.25$ to attain the single floating point precision limit for the visibility misfit. When a look–up table is used, the sampling rate is advised to be 10^6.

If we look back to the degridding process described in the Cotton–Schwab CLEAN in Section 1.4, this was introduced to replace the convolution in the image plane by a multiplication

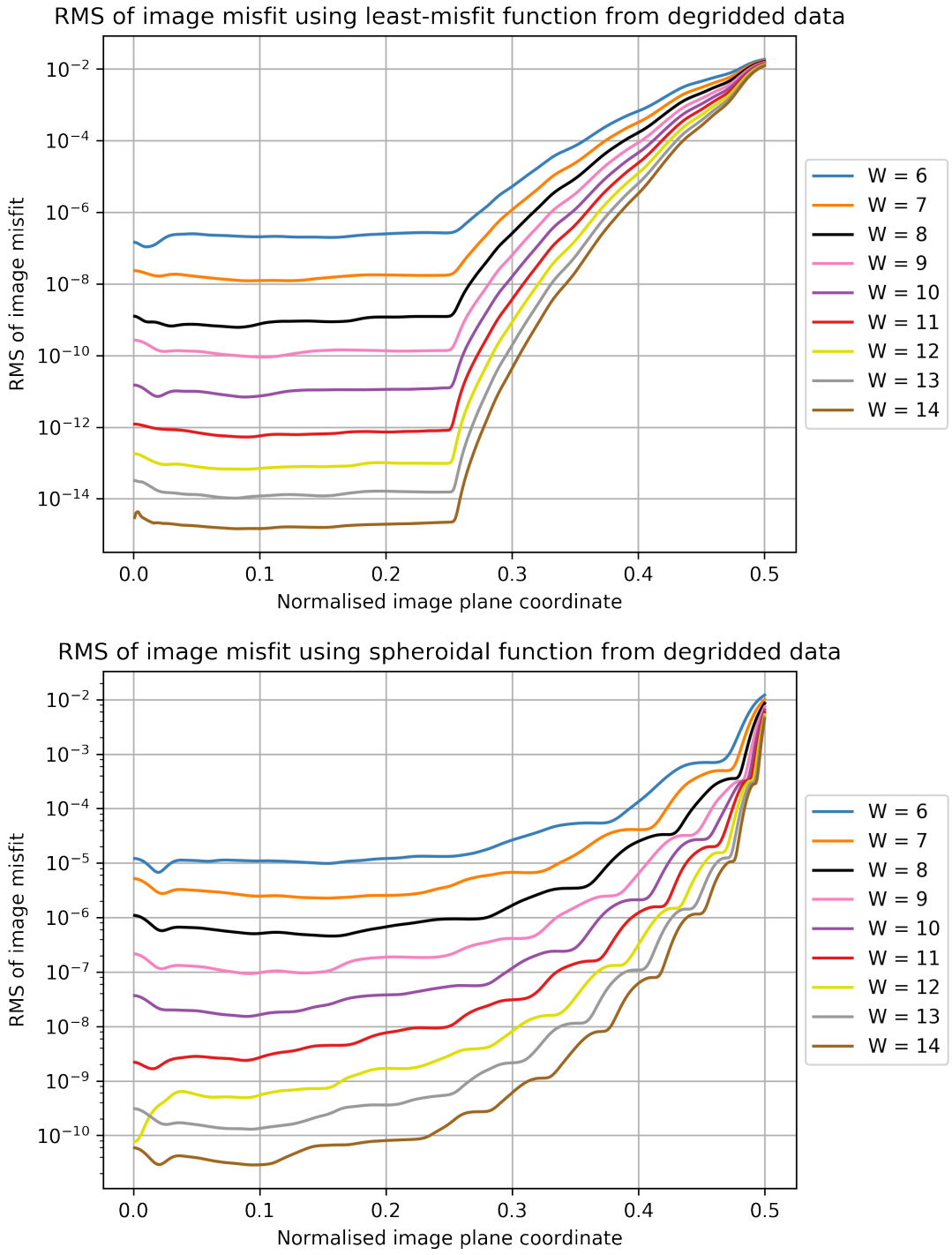


Figure 3.46: RMS results of the image misfit between the DFT dirty image constructed from simulated visibilities and the FFT dirty image constructed from degridded visibilities. The least–misfit gridding function is used in the upper plot, and the spheroidal function is used in the lower plot.

on the visibility plane, so as to reduce the computational cost. The calculated visibility residual is then to be gridded and the inverse FFT applied to construct residual maps for new rounds of minor cycles. We therefore need to conduct experiments so as to investigate the difference between the FFT dirty images made from the degridded visibilities and the DFT dirty images made from the original visibilities. If the difference is small enough to be neglected, the introduction of major cycles will not be viewed as introducing extra error but merely reducing the computational cost with no impairment in performance.

The simulated visibilities used here are the same single point–source data made earlier. The corresponding image model is a 500 by 500 all–zero matrix other than at $(250, 250)$, where the point source lies. In the experiment, we first construct the DFT dirty image from the simulated data. We shall also construct the FFT dirty image from the degridded visibility data, which is obtained from the degridding process from the pre–made image model. The image model is first corrected by the correcting function on both axes, then zero–padded to 1000×1000, followed by an FFT and a correlation to give the degridded visibilities. The RMS value of the image misfit between the two dirty images is plotted against the normalised image plane coordinate in Figure 3.46. The least–misfit gridding function is used in the upper plot, and the spheroidal function is used in the lower plot.

The two plots are consistent with the RMS results in Figures 3.30 and 3.31, where the image misfit is different and refers to the DFT and FFT dirty images made from the same simulated point–source data. In view of the different visibility data used for the two types of FFT dirty image, we expect the FFT image made from the degridded visibility data to be less accurate than that made from the original visibilities. The RMS results in Figure 3.46 confirm this intuition: in the bottom plot, the RMS results are noticeably larger than in Figure 3.31, while in the top plot the difference still exists in comparison with Figure 3.30, but is less obvious.

Figure 3.46 shows also that the least–misfit gridding function causes the image misfit to be at least 100 times smaller than when the spheroidal function is used, for fixed W. In conclusion, it is recommended that the least–misfit gridding function is used during the degridding process. As discussed earlier, the values $W = 7$ and $x_0 = 0.25$ are advised; these make the image misfit to attain the single precision limit without further ado.

3.7 Conclusions

In this chapter, we proposed a new set of gridding functions based on the criterion that the difference between the DFT and FFT dirty images should be minimised. This is based in turn on the conclusion drawn in Chapter 2, where the results of source extraction from dirty images are more accurate than on CLEANed images. As a result, accurate dirty images can potentially

provide more accurate information about the radio sky, and high–precision dirty images will benefit future applications that work directly with dirty images.

We began this chapter with a survey of historical gridding functions, such as the pillbox and truncated sinc function. After a brief introduction to the prolate spheroidal wave functions, which were designed and used to suppress aliasing effects, we set out the theory behind our least–misfit gridding functions. We then specified the corresponding optimisation and implementation procedures. Because the matrix $\boldsymbol{A}$ is ill-conditioned, optimisation fails when the gridding function support width W exceeds 11. In this case, $W > 11$, a more robust optimisation via a different matrix $\boldsymbol{B}$ has been proposed.

Theoretical and (numerical) experimental comparisons between the spheroidal functions and the least–misfit gridding functions were undertaken. In the theoretical work, we compared the aliasing suppression ratio and the theoretical map error function $\ell(x)$. We reach two conclusions:

- With W fixed, the least–misfit gridding function has a steadier and smaller aliasing suppression ratio than the spheroidal function.
- With W fixed, the least–misfit gridding function with $x_0 = 0.25$ has a steadier and smaller map error function $\ell(x)$ than the spheroidal function.

In the experimental comparison, we compared the RMS value of the image misfit and normalised aliasing brightness using the simulated VLA data. We found that the least–misfit gridding function outperforms the spheroidal function in both aliasing suppression and image misfit minimisation by a factor of at least 100. These findings are consistent with the theoretical analysis.

In summary, the least–misfit gridding function was chosen by minimising the misfit between the DFT and FFT dirty images. Also, the least–misfit gridding function gives better aliasing suppression than the spheroidal function with the same W, even though the spheroidal function was designed to suppress aliasing effects. Essentially, aliasing is a part of the image misfit, and there is therefore no need to consider its suppression separately.

We also considered the computational cost and degridding process. We recommend the use of the least–misfit gridding function with $W = 7$ and $x_0 = 0.25$, so as to achieve the desired accuracy of the single precision limit with a comparable and small computational cost in both the gridding and degridding processes. If a look–up table is used, nearest–neighbour method is preferred. The sampling rate should be not less than 10^6 so as to attain the single precision limit while using least–misfit function with $W = 7$ and $x_0 = 0.25$. It is noted that even if a single precision is not demanded, the least–misfit function can achieve at least 100 times better image accuracy in both gridding and degridding process than spheroidal function with the same W and x_0. With the same computational cost, therefore the least–misfit function is preferred.

In the future, we propose to compare the accuracy of BaSC results from dirty images made from the same visibility data but with different gridding functions. We expect that use of the least–misfit gridding function will lead to the more accurate finding of sources. Second, since the least–misfit gridding function greatly improves the accuracy in both the gridding and degridding processes, we intend to explore the improvements in efficiency that the least–misfit gridding function would bring to the image reconstruction process using further numerical experiments. Finally, we shall seek to implement the least–misfit gridding function as a gridding function option in one of the main-stream data reduction pipelines.

Chapter 4

Accurate approaches to wide–field imaging

4.1 Introduction

Let us return to the relation between the sky brightness distribution $I(l, m)$ and the visibility data $V(u, v, w)$ written in Equation 1.1:

$$V(u,v,w) = \int\int \frac{\mathrm{d}l\mathrm{d}m}{\sqrt{1-l^2-m^2}} I(l,m) \exp[-i2\pi(ul + vm + w(\sqrt{1-l^2-m^2} - 1))]$$

In previous chapters, the field–of–view (FoV) of our test data was taken to be small, so that $n = \sqrt{1-l^2-m^2} \approx 1$. Hence, $w(n-1)$, the so–called 'w–term' (Perley 1989), can be ignored. By doing so, we are able to apply a two–dimensional inverse Fourier transform to the observed visibility data so as to obtain a dirty image, and then choose a deconvolution algorithm such as CLEAN (Högbom 1974; Clark 1980; Schwab & Cotton 1983; Cornwell 2008) or Maximum Entropy method (Gull & Daniell 1978; Gull & Skilling 1984b; Gull 1989; Skilling 1989) to reconstruct the sky image from it. These procedures were explained in Chapter 1.

If the FoV increases, however, the two–dimensional transform is no longer applicable. We now give an example to show the erroneous effects that arise if the w–term is neglected.

Figure 4.1 shows two dirty images of size 900×900 pixels constructed from a simulated 22-minute VLA B–array data. The data were simulated in CASA at 74 MHz, containing 34 point sources whose fluxes and positions are specified in Appendix C. The hour angle changes from $-5h$ to $5h$ by an interval of $1h$ every 2 mins. Since multiple hour angles are used, the values of w are significant in the data, so that this is a wide–field imaging problem. The two images

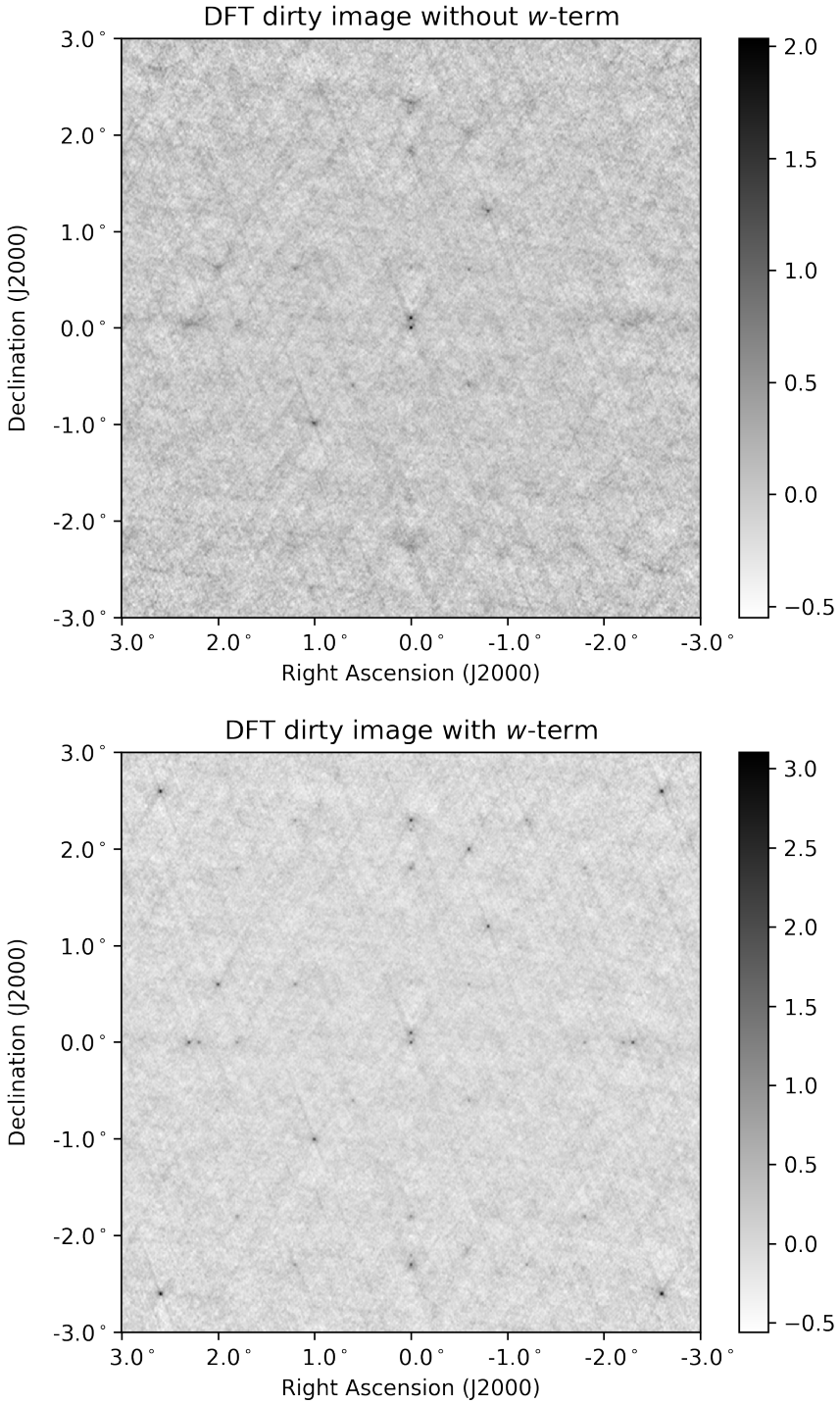


Figure 4.1: The upper dirty image arises from a direct inverse two–dimensional transform neglecting the w–term, while the lower image is based on Equation 1.1.

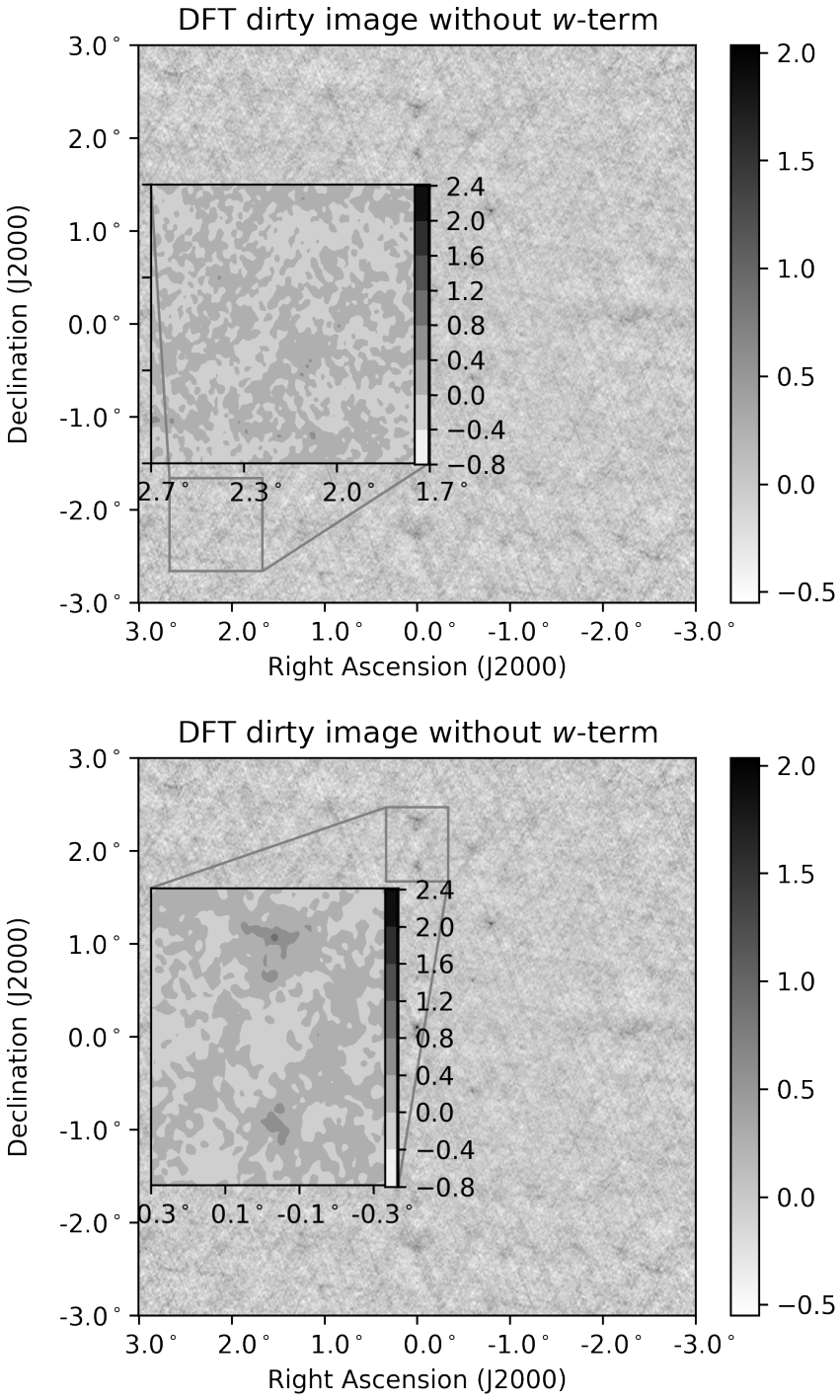


Figure 4.2: Both images are the same DFT dirty image without the w–term taken into account, but having different zoomed–in windows shown in contours.

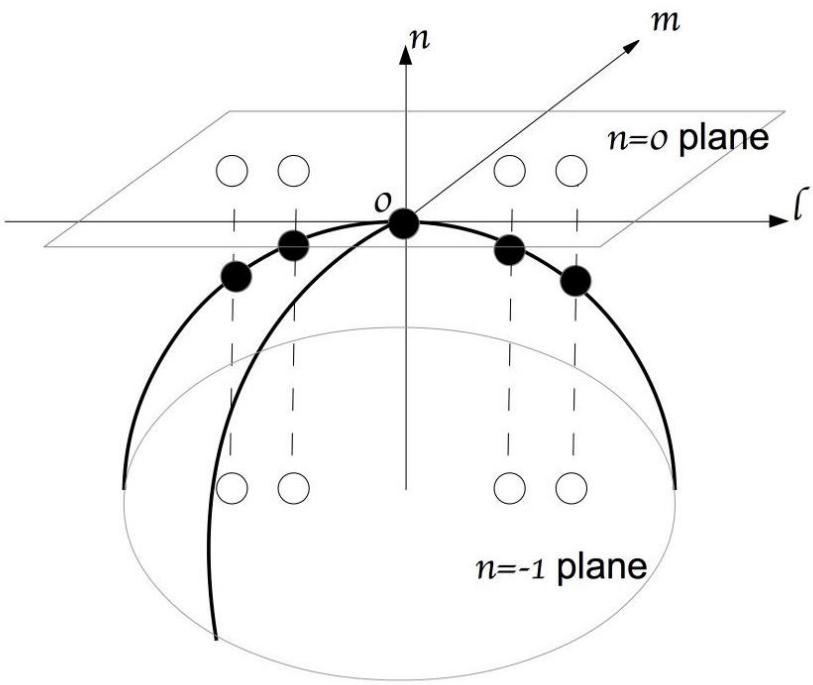


Figure 4.3: The three–dimensional image coordinate system in radio interferometry. On the celestial sphere are shown 5 sources, and the central source is on the tangent plane $n = 0$ but the others are not. The targeted sky brightness is the projection of the celestial sphere onto the tangent plane.

are both 6 degrees across, with a pixel size of 24 arcsec. The upper image is constructed by a direct inverse two–dimensional Fourier transform with $w = 0$, and the lower image is based on Equation 1.1 with the w–term included. Primary beam correction has been applied.

We clearly see multiple point sources in the lower image, while in the upper image we discern only two distinct sources in the middle. The largest flux level in the lower image is also larger than that in the upper image. In addition, the four corner sources, which have the largest fluxes (3 Jy) among the 34 sources, cannot be seen in the dirty image constructed without the w–term, see the zoomed–in window in the top image in Figure 4.2. At the same time, many of the sources are blurred; see the zoomed–in window in the bottom image in Figure 4.2, where the two point sources are blurred.

4.1.1 Three–dimensional image coordinates

We introduce the three–dimensional image coordinates used in this chapter. For a given radio interferometer, we write the phase–tracked visibilities $V'(u, v, w)$ as a phase rotation of the raw

visibilities $V(u, v, w)$ (Perley 1989) in order to simplify the mathematics:

$$V'(u, v, w) = \exp(i2\pi w)V(u, v, w) \tag{4.1}$$

As a result, Equation 1.1 can be written as:

$$V'(u, v, w) = \int\int \frac{\mathrm{d}l\mathrm{d}m}{n} I(l, m) \exp[-i2\pi(ul + vm + wn)]$$

This phase rotation in the visibility plane reflects a unit downward shift of the n–plane (that is, the plane pierced perpendicularly by the n-axis). Although n should normally be in the range $(0, 1)$ by definition, it is shifted into the range $(-1, 0)$, as shown in Figure 4.3. The plane $n = 0$ represents the 'tangent plane'. In the schematic diagram there are 5 sources demonstrated on the celestial sphere; only one source is in the tangent plane $n = 0$.

The top map in Figure 4.3 is a dirty image made on the tangent plane, and the bottom graphical representation shows the dirty image of the three–dimensional celestial sphere projected onto the tangent plane. Throughout this chapter, the three–dimensional image coordinate system is used if not specified. For simplicity, we drop the prime and use V to denote V'. Hence we may rewrite the relation between the visibility data and brightness distribution as:

$$V(u, v, w) = \int\int \frac{\mathrm{d}l\mathrm{d}m}{n} I(l, m) \exp[-i2\pi(ul + vm + wn)] \tag{4.2}$$

4.1.2 The Fresnel number

The origin of the w–term is geometric, as explained in (Perley 1989). The left graphic in Figure 4.4 shows two antennas on the same uv plane. We suppose for simplicity that they have the same value of v. In this way, the phase difference between signals received by these two antennas at angle θ can be written as $2\pi u \sin\theta$. This phase difference changes to $2\pi(u \sin\theta + w_0 \cos\theta)$ when a non–coplanar array is used; see the right graphic in Figure 4.4, where w_0 is the w–component of the geometric difference of the antennas. This extra phase difference gives rise to phase errors if it is neglected. The phase error can in general be written as:

$$\phi_w = 2\pi w(\sqrt{1 - l^2 - m^2}) \sim \pi w(l^2 + m^2) \tag{4.3}$$

The small angle approximation is used here.

Based on this formula, as set out in Cornwell & Perley (1992), this phase error grows linearly with the size of the array (the value of w, in wavelength) and quadratically with distance from the phase centre (dependent on l and m). In Figure 4.1, for example, the four sources at each corner of the lower image cannot be seen in the upper image due to the large values of l, m, whereas the two point sources around the phase centre can be seen in both images because

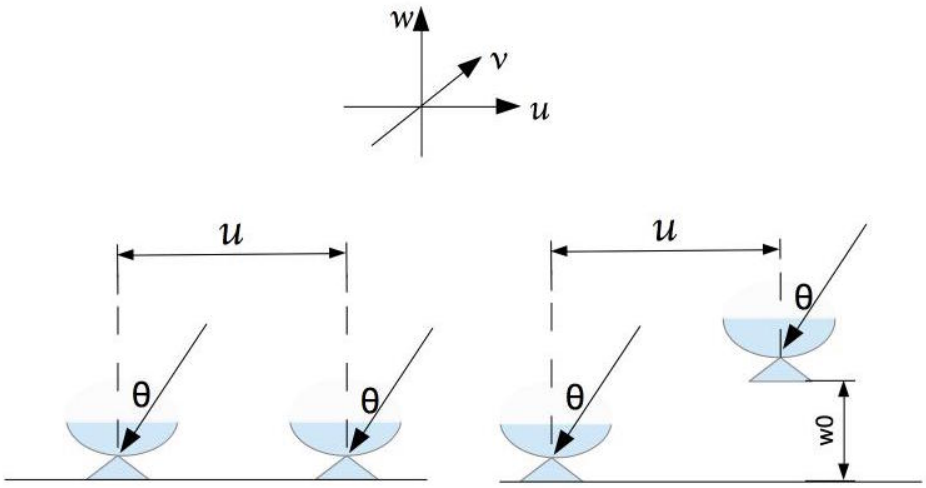


Figure 4.4: The left figure shows two antennas on the same uv–plane receiving radiation emitted from a remote source, and the right graphic shows a non–coplanar array with a difference w_0 in the w–axis. The phase error caused by this difference in w causes blurring of the sources on the image plane if the w–term is ignored.

l, m are sufficiently small. In the present work this blurring effect will be termed the 'non–coplanar effect', and we shall refer to imaging methods that deal with the non–coplanar effect as 'wide–field imaging methods' or 'wide–field imaging algorithms'.

Assuming that w is maximised at $\frac{B_{\max}}{\lambda}$, and $l^2 + m^2$ is maximised at $(2\lambda/D)^2$ (Cornwell & Perley 1992), we obtain

$$\phi_w \sim \frac{\lambda B_{\max}}{D^2} \tag{4.4}$$

where $B_{\max}$ is the longest baseline and D is the diameter of the antenna element. If the phase error is much smaller than 1, or

$$\frac{\lambda B_{\max}}{D^2} \ll 1 \tag{4.5}$$

then the phase error can be omitted and the two–dimensional Fourier inversion is applicable. The Fresnel number N_F was introduced as a quantitative measure of whether non–coplanar effects should be taken into account (Cornwell & Perley 1992; Cornwell et al. 2008):

$$N_F = \frac{D^2}{\lambda B_{\max}} \tag{4.6}$$

If $N_F < 1$, the non–coplanar effects should be carefully removed. Extra care should be taken with low frequency observations, especially with long baselines, which will correspond to small Fresnel numbers.

Since the VLA extended its low frequency coverage to the P band (330 MHz) and the 4 band (74 MHz), its observational data has furthered much scientific research in fields such as supernova remnants (Bietenholz et al. 1997), and radio halos and relics (Kassim et al. 2001). With the construction of low frequency interferometers such as the Giant Metrewave Radio Telescope (GMRT) (Swarup et al. 1991), the Murchison Widefield Array (MWA) (Tingay et al. 2013), the Long Wavelength Array (LWA) (Ellingson et al. 2009), the Low Frequency Array (LOFAR) and the Low-Frequency Aperture Array (LFAA) (de Vaate et al. 2011) of the Square Kilometre Array, progress is expected in areas of science such as the epoch of reionization and deep extragalactic surveys. These radio interferometers face an identical challenge of wide–field imaging with small Fresnel numbers, and it is therefore important to develop good wide–field imaging methods. The computational load is a further concern due to the increasing amount of incoming data that will be involved.

4.1.3 Planar arrays

Even when observed by planar arrays, visibility data with a significant w–term may cause serious non–coplanar effects. This occurs when the phase centre of the observation is not normal to the plane of the array. Wide–field imaging is then needed. We first focus on planar arrays, before considering more general wide-field imaging methods.

We simulated a simple VLA snapshot generated by the A–configuration of just 351 baselines. We generate this single-channel data via a degridding process using our least–misfit gridding function with $W = 7$ and $x_0 = 0.25$. In this way the source fluxes are not attenuated by the primary beam. It contains the same 34 point sources shown in Figure 4.1[a]. To ensure a sizeable value of n at the image corners, we rescale the image range to $[-\pi/10,\pi/10]$ by changing the observation frequency. We obtain $n \approx -0.10$ at the corners of the 900 by 900 map, with the ranges of l and m both being $2\sin(\pi/10) \approx 0.62$. The effects of non–zero w are therefore considerable, despite the VLA being an essentially planar array. A good way to quantify the non-planarity is to examine the singular values of the (u, v, w) set. We find singular values of $(251.22, 232.04, 0.53)$, of which the singular value 0.53 (very close to zero) indicates that (u, v, w) has a linear dependence. By plotting all w values against the v values in Figure 4.5, we find that the values of v and w of the observations are almost perfectly correlated:

$$w = -0.6737v; R^2 = 0.9998 \tag{4.7}$$

[a] Apart from the change of angle between the two central sources, so that these sources can be easily distinguished even with the bad *uv*-coverage.

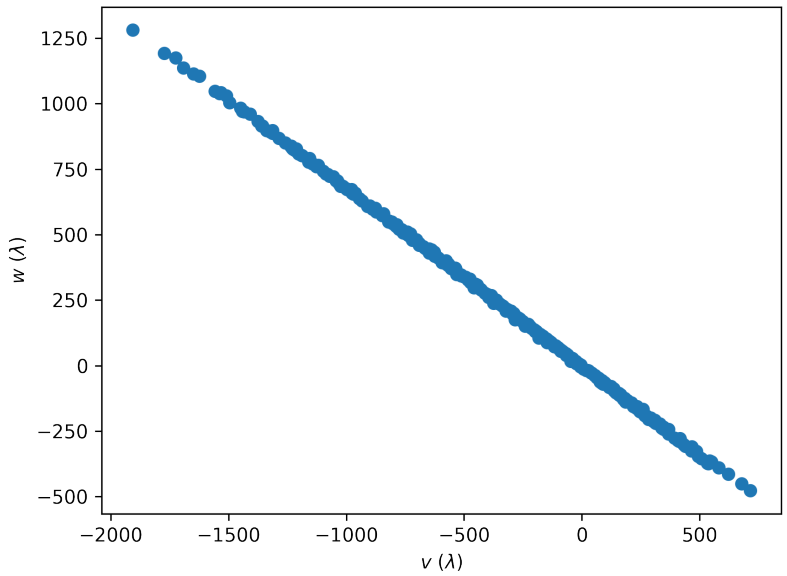


Figure 4.5: w values of the observation plotted against v values in the same VLA snapshot. Both w and v are in units of wavelength.

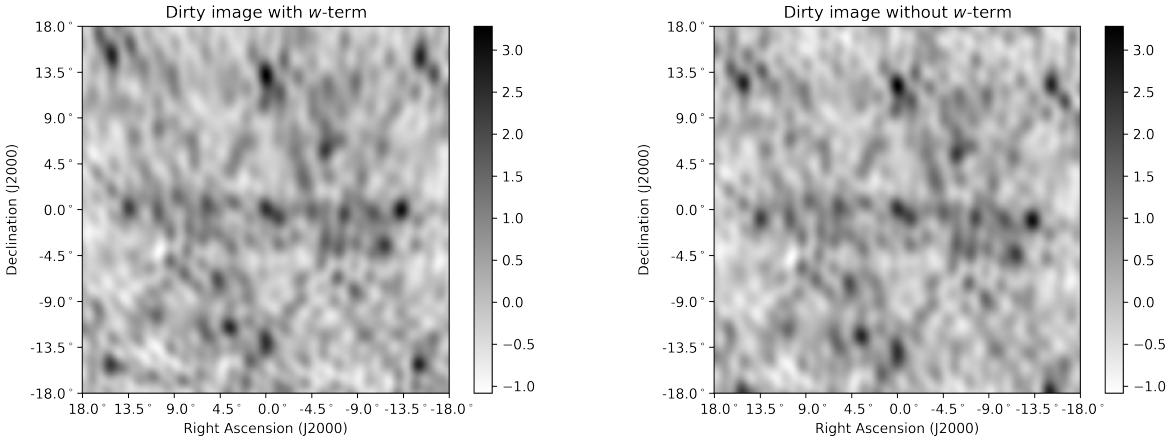


Figure 4.6: Left: dirty image on the celestial sphere made from the VLA snapshot data; Right: tangent plane dirty image. The sources in the tangent plane are displaced by a small amount to negative declination.

where R is the correlation coefficient, very near to 1. This also indicates that the array is almost perfectly planar.

The dirty image on the celestial sphere made from the snapshot data is shown in the left image of Figure 4.6. The tangent-plane image produced by a direct two–dimensional Fourier transform is shown on the right in Figure 4.7, where the w–term is omitted. If the array were non–coplanar then we would expect sources away from the phase centre to become blurred,

but this does not happen for a planar array. Upon comparing the two images, we see that the sources are simply displaced to negative declination by a small amount.

This effect is explained as follows. Consider a point source with a flux F at (l, m, n), which produces complex visibilities

$$V_k = F \exp[-i2\pi(lu_k + mv_k + nw_k)] \pm \sigma_k$$

. Here, σ_k is the standard deviation of the noise, which is assumed to be Gaussian. The dirty image made from the coplanar arrays can now be written as

$$I_D(\boldsymbol{x}) \equiv \mathfrak{R}\left\{\sum_k \frac{V_k}{\sigma_k^2} \exp[i2\pi(lu_k + mv_k + nw_k)]\right\} \sigma^2; \frac{1}{\sigma^2} \equiv \sum_k \frac{1}{\sigma_k^2} \tag{4.8}$$

In the present case with $w_k = \alpha v_k$, the w–term results simply in a shift of the declination of the sources by $-\alpha n$. For example, the four corner sources visible in the image at $(l, m) = (\pm 0.26, \pm 0.26)$ have $n = -0.07$[b]. Since $\alpha = -0.67$, the displacement for these sources should be approximately $m = -0.05$, or $2.86°$. The displacement effect in Figure 4.6 is confirmed to be consistent with the theory.

This specific case has $w = \alpha v_k$, given a planar array with $w_k = \alpha u_k + \beta v_k$, i.e. a linear function of (u, v). The above analysis generalises straightforwardly, so the displacement in (l, m) becomes $(-\alpha n, -\beta n)$.

We further decrease the observation frequency and rescale the 34 sources within a wider field; see Figure 4.7. Both maps are dirty images on the celestial sphere, but with differing FoV. The FoV of the left dirty image is scaled up to the range $-\pi/5$ to $\pi/5$, and the right image is in the range $(-\pi/3, -\pi/3)$. Both images are projections of the celestial sphere onto the tangent plane. In the right map, the corners of the map are already outside the celestial sphere, and are therefore left blank. The beam shape is foreshortened in the outer parts of the map, because we are viewing a projection of the celestial sphere.

We also see that the bright sources in the corners are better resolved at the bottom than those at the top. This is particularly obvious in the right map. It is because the resolution obtained with the planar array varies across the sky. Specifically, the beam width varies inversely with the cosine of the angle from the normal to the plane of the array, so that there is no resolution at the plane of the array.

The flux levels of the top and bottom corner sources are nevertheless the same, at 3 Jy, although the top corner sources are much more extended. This indicates that the dirty image at any position represents the flux that a source would have if there were one at that point; consequently, it does not depend on the resolution.

[b]For these values of (l, m) the approximation $n = -\frac{1}{2}(l^2 + m^2)$ is very nearly adequate.

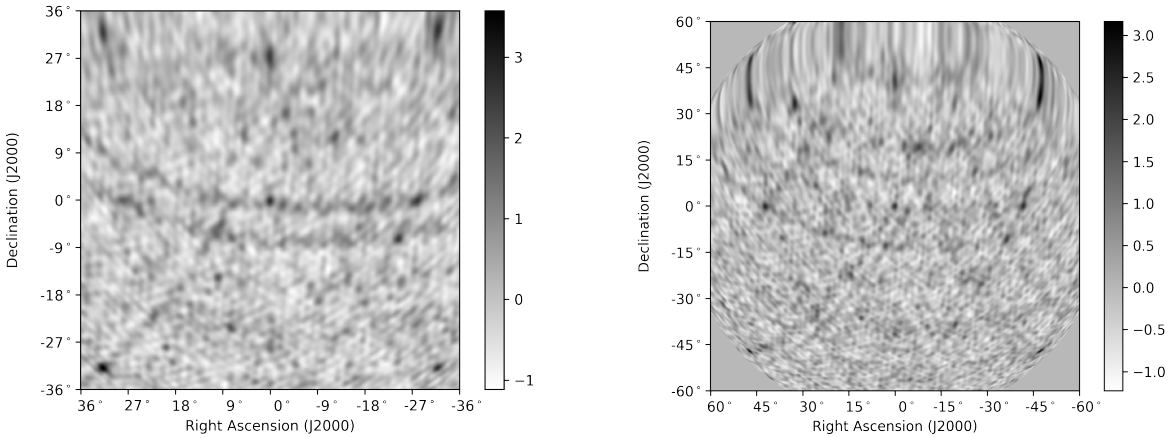


Figure 4.7: Left: Dirty image constructed from simulated VLA snapshot data using an angular field of $\pm\pi/5$ in both dimensions. (b) Dirty image made from other simulated snapshot data using an angular field of $\pm\pi/3$ in both dimensions.

Most radio interferometers have non–coplanar arrays, however. A general method for handling the w–term is needed. This is particularly important as more low-frequency observations are conducted with the new generation of radio interferometers. Many methods have been proposed, such as the three–dimensional Fourier transform (Perley 1989), warped snapshot (Perley 1989), the polyhedron imaging method (Perley 1989; Cornwell & Perley 1992), W–Projection (Cornwell et al. 2003, 2005, 2008), W–Stacking (Humphreys & Cornwell 2011; Offringa et al. 2014) and W-Snapshot (Cornwell et al. 2012).

In this chapter, we first introduce some of the existing wide field imaging methods, in Section 4.2. We then propose an improved W–Stacking method, in Section 4.3. After that, a novel approach called 'N-Faceting' is introduced in Section 4.4. The performance of the two proposed methods will be demonstrated using simulated data in Section 4.5. We also compare the improved W–Stacking method with W–Projection implemented in `CASA`, and W–Stacking implemented in `WSClean`. The computational comparison between the original and improved W–Stacking methods is discussed in Section 4.6, and conclusions are drawn in Section 4.7.

4.2 Some existing wide–field imaging methods

In this section we introduce the three–dimensional Fourier transform method (Perley 1989), the polyhedron imaging method (Perley 1989; Cornwell & Perley 1992), W–Projection (Cornwell et al. 2003, 2005, 2008) and W–Stacking (Humphreys & Cornwell 2011; Offringa et al. 2014) methods.

4.2.1 3-D Fourier Transform method

Perley (1989) first introduced the three–dimensional Fourier Transform method to recover the sky brightness distribution when the w–term cannot be ignored. From Equation 4.2, we obtain:

$$V(u,v,w) = \int\int\int \mathrm{d}l\mathrm{d}m\mathrm{d}n \frac{I(l,m)\delta(\sqrt{1-l^2-m^2}-n)}{\sqrt{1-l^2-m^2}} \exp\left[-i2\pi(ul+vm+wn)\right] \quad (4.9)$$

The relation between $V(u,v,w)$ and $I(l,m)$ is quite similar to the three dimensional Fourier transform. Hence, $F(l,m,n)$ is introduced as an inverse three–dimensional Fourier transform of $V(u,v,w)$:

$$F(l,m,n) = \int\int\int \mathrm{d}u\mathrm{d}v\mathrm{d}w V(u,v,w) \exp\left[i2\pi(ul+vm+wn)\right], \quad (4.10)$$

The term $F(l,m,n)$ is defined as the 'three–dimensional volume' by Perley (1989). The relation between the three–dimensional volume and the sky brightness $I(l,m)$ can be derived as follows:

$$\begin{aligned}
F(l,m,n) &= \int\int\int \mathrm{d}u\mathrm{d}v\mathrm{d}w \left\{ \int\int \mathrm{d}l'\mathrm{d}m' \frac{I(l',m')}{\sqrt{1-l'^2-m'^2}} \exp\left\{2i\pi(ul'+vm'+w\sqrt{1-l'^2-m'^2})\right\}\right\} \\
&\exp\left[i2\pi(ul+vm+wn)\right] \\
&= \int\int \mathrm{d}l'\mathrm{d}m' \left\{ \int\int\int\int \mathrm{d}u\mathrm{d}v\mathrm{d}w \frac{I(l',m')}{\sqrt{1-l'^2-m'^2}} \exp\left[-i2\pi u(l'-l)\right] \exp\left[-i2\pi v(m'-m)\right] \right. \\
&\left. \exp\left[-i2\pi w(\sqrt{1-l'^2-m'^2}-n)\right]\right\} \\
&= \int\int \mathrm{d}l'\mathrm{d}m' \frac{I(l',m')}{\sqrt{1-l'^2-m'^2}} \delta(l'-l)\delta(m'-m)\delta(\sqrt{1-l'^2-m'^2}-n) \\
&= \frac{I(l,m)\delta(\sqrt{1-l^2-m^2}-n)}{\sqrt{1-l^2-m^2}}
\end{aligned}$$

To analyse the relation between $F(l,m,n)$ and $I(l,m)$, we look at the five sources on the unit celestial sphere shown in Figure 4.3. $F(l,m,n)$ represents a three dimensional image cube which includes these sources. For each set of (l,m), $I(l,m)$ is the projection of $F(l,m,n)$ onto the $n = \sqrt{1-l^2-m^2}$ plane. If the visibility data is the complete sample of the radio sky then $F(l,m,n)$ can be obtained by a direct inverse three–dimensional Fourier transform. As a result of introducing the three–dimensional sampling function $S(u,v,w)$, the result of the inverse three–dimensional Fourier transform becomes:

$$\begin{aligned}
\widehat{F}(l,m,n) &= \mathscr{F}_{3D}^{-1}\{V(u,v,w)S(u,v,w)\} \\
&= F(l,m,n) * B^{3D}(l,m,n)
\end{aligned}$$

where $\mathscr{F}_{3D}^{-1}$ denotes the three–dimensional inverse Fourier transform, so that $B^{3D}(l,m,n)$ is the three–dimensional inverse Fourier transform of the sampling function $S(u,v,w)$, defined as the

three–dimensional dirty beam:

$$B^{3D}(l,m,n) = \int\int\int \mathrm{d}u\mathrm{d}v\mathrm{d}w S(u,v,w)\exp\left[i2\pi(ul+vm+wn)\right] \tag{4.11}$$

Also, the convolution refers to a three–dimensional convolution. From this point we can use deconvolution methods to reconstruct $F(l,m,n)$ from $\widehat{F}(l,m,n)$.

In summary, the basic steps in recovering the brightness distribution $I(l,m)$ using the 3D Fourier Transform method are:

1) Apply an inverse three–dimensional Fourier transform to the visibilities $V(u,v,w)$ to obtain $\widehat{F}(l,m,n)$.
2) Use deconvolution algorithms to reconstruct the three–dimensional volume $F(l,m,n)$ by deconvolving the three–dimensional dirty beam $B^{3D}(l,m,n)$ from the $\widehat{F}(l,m,n)$.
3) Project the result $F(l,m,n)$ onto a chosen plane (usually the tangent plane $n=0$) to obtain the desired sky brightness $I(l,m)$ according to

$$F(l,m,n) = \frac{I(l,m)\delta(\sqrt{1-l^2-m^2}-n)}{\sqrt{1-l^2-m^2}}$$

.

Instead of the direct Fourier transform, we can use FFT so as to save computational time, where gridding is essential to assign visibility values to three dimensional grids. If the n axis is undersampled, however, this will result in severe aliasing in the FoV (Perley 1989).

The number of planes along the n axis, denoted N_{planes}, must be determined. Following Perley (1989), we show the unit celestial sphere in Figure 4.8, where the FoV is within the angle 2θ, which is limited by $\theta \approx \lambda/D$. On the celestial sphere, the range of cell numbers on the n axis runs from $-(1-\cos\theta)$ to 0. Here, n should be sampled as densely as l and m, namely $\delta n \leq \lambda/2B_{\mathrm{max}}$. Hence the plane number N_{planes} can be calculated as the number of samples needed on the n axis:

$$\begin{aligned} N_{\mathrm{planes}} &\geq (1-\cos\theta)/(\lambda/2B_{\mathrm{max}}) \\ &\approx (\theta^2/2)/(\lambda/2B_{\mathrm{max}}) \\ &= \frac{B_{\mathrm{max}}}{\theta^2\lambda} \\ &\approx \lambda B_{\mathrm{max}}/D^2 \end{aligned}$$

The number of planes along the n axis is therefore constrained by:

$$N_{\mathrm{planes}} \geq \lambda B_{\mathrm{max}}/D^2 \tag{4.12}$$

According to the formula, for a certain observation, if $N_{\mathrm{planes}} \leq 1$, a standard two–dimensional Fourier inversion can safely be applied. Otherwise, the w–term demands carefully handling.

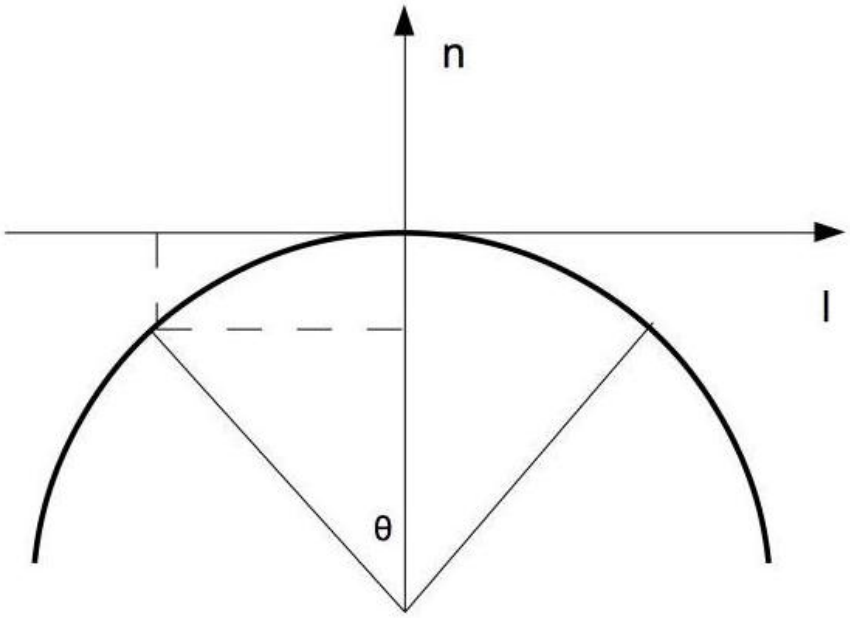


Figure 4.8: The field of view is within the angle 2θ. On the unit celestial sphere, the range of cell numbers on the n axis runs from 0 to $1 - \cos\theta$. The plane number N_{planes} can be calculated as the number of samples needed on the n axis.

Once N_{planes} has been determined, the three-dimensional Fourier transform can be performed by FFT along all three axes.

In summary, the $3D$ Fourier Transform method is mathematically straightforward and analytically precise. But it has been used only sparingly because of its high computational cost and the complex deconvolution process.

4.2.2 Polyhedron imaging method

The polyhedron imaging method, or image–plane faceting, was first introduced by Perley (1989) as an alternative to the 3D Fourier transform method at much lower computational cost. The basic idea is to divide the wide FoV into several small FoVs, or *facets*, in which the two–dimensional Fourier relation still applies. The desired sky image then can be obtained by stitching the reconstructed sky images of each facet altogether. The method was implemented and demonstrated using VLA data observing at 327 MHz on 3C84 in Perley & Cornwell (1991). It has also been utilised in the imaging process from many low-frequency wide–field observations, including LaRosa et al. (2000),Nord et al. (2004), and Whittam et al. (2017).

Figure 4.9 shows a schematic diagram of the polyhedron imaging method. The field of

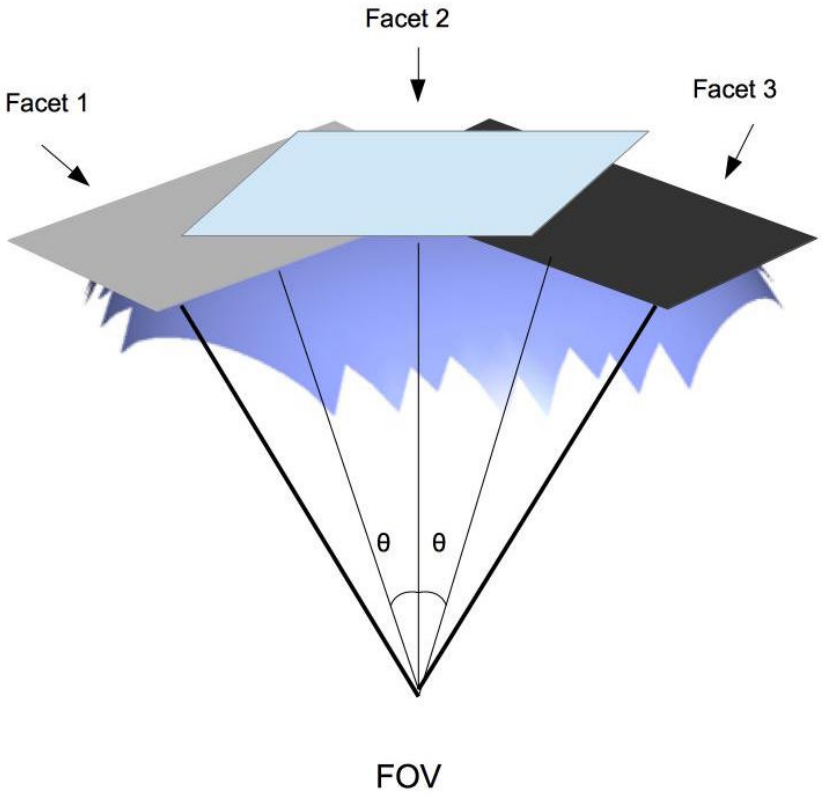


Figure 4.9: A schematic example of polyhedron imaging. The field of view is shown within the two bold black lines, which can be divided into three smaller fields of view, or 3 facets: Facet 2 is on the tangent plane, and the others are displaced by an angle $\pm\theta$ with different reference centres. In practice, usually more than three facets are needed.

view is shown within the two bold black lines, which can be divided into three smaller fields of view, or facets; Facet 2 is on the tangent plane, and the other two are displaced by an angle $\pm\theta$, with different reference centres. In practice, usually more than three facets are needed.

The phase reference centres are different for different facets, so if the phase reference centre for a certain facet is (l_0, m_0, n_0), the visibilities $V(u, v, w)$ must be phase–shifted to become:

$$V_0 = V(u, v, w) \exp\{i2\pi[u(l - l_0) + v(m_0 - m) + w(n_0 - n)]\} \tag{4.13}$$

The dirty image of this facet is then obtained by applying a two–dimensional inverse FFT to the phase–shifted and gridded visibility data V_0. Hence, a deconvolution algorithm can be used to reconstruct the sky images of this particular facet.

Perley (1989) pointed out that the number of facets N_{facets} can be determined by:

$$N_{\text{facets}} = \frac{2\lambda B_{\max}}{fD^2} \tag{4.14}$$

where the factor f should be carefully chosen: $f = 1$ is for critical sampling, and $f < 1$ holds for high dynamic range.

In summary, the steps for image–plane faceting are:

1) Determine the number of facets needed.
2) For each facet, apply a two–dimensional inverse FFT to the phase–shifted and gridded visibilities relating to its reference point, so as to obtain its corresponding dirty image.
3) Use a deconvolution method such as CLEAN to reconstruct the sky image for each facet.
4) Project all the reconstructed facets onto the tangent plane to give the final image.

In Step 3, if CLEAN is used, the minor cycle will be applied in each facet image with local dirty beams, whereas in the major cycle, the degridded visibilities from each facet will be subtracted to form an overall residual image.

An alternative faceting method is called 'visibility–plane faceting' or 'uv–plane faceting' (Sault et al. 1999). Visibility phase shifting and image summing involved in the image–plane faceting method are both linear operations, and the equivalent operation in the visibility–plane faceting can be summarised as follows:

1) Determine the number of uv–facets needed.
2) For each facet, grid visibilities onto the same grids but with different corresponding phase centres.
3) Apply a two–dimensional FFT to the visibilities on grids, to obtain a single dirty image.
4) Deconvolve the desired sky image from a single dirty beam as chosen.

During the deconvolution process, instead of using multiple local beams on the facets, only one dirty beam is needed during the visibility–plane faceting; this saves computational time (Cornwell et al. 2008). In contrast to image–plane faceting, visibility–plane faceting is not affected by radio emissions across multiple facets, or edge errors due to overlapping image–plane facets. It also can be combined with the W–Projection method for very large field of view.

4.2.3 W–Projection

4.2.3.1 Overview

The W–Projection algorithm was developed based on the idea that we can still take advantage of the two–dimensional Fourier transform by projecting w onto the $w = 0$ plane (Cornwell et al.

2003, 2005, 2008). From Equation 1.1, the w–term can be separated as[c]:

$$V(u,v,w) = \int\int \mathrm{d}l\mathrm{d}m \frac{I(l,m)}{\sqrt{1-l^2-m^2}} \exp\left[-i2\pi(ul+vm)\right] \exp\{-i2\pi w[\sqrt{1-l^2-m^2}-1]\} \tag{4.15}$$

The separated exponential component containing the w–term is defined as $G(l,m,w)$:

$$G(l,m,w) = \exp\{-i2\pi w[\sqrt{1-l^2-m^2}-1]\} \tag{4.16}$$

Equation 4.15 can therefore be written as:

$$V(u,v,w) = \int\int \mathrm{d}l\mathrm{d}m \frac{I(l,m)}{\sqrt{1-l^2-m^2}} \exp\left[-i2\pi(ul+vm)\right] G(l,m,w) \tag{4.17}$$

which can be further simplified as:

$$\begin{aligned} V(u,v,w) &= \mathscr{F}_{2D}\left\{\frac{I(l,m)}{\sqrt{1-l^2-m^2}} G(l,m,w)\right\} \\ &= \mathscr{F}_{2D}\left\{\frac{I(l,m)}{\sqrt{1-l^2-m^2}}\right\} * \mathscr{F}_{2D}\{G(l,m,w)\} \\ &= V(u,v,w=0) * \mathscr{F}_{2D}\{G(l,m,w)\} \\ &= V(u,v,w=0) * \tilde{G}(u,v,w) \end{aligned} \tag{4.18}$$

This shows that visibilities with non–zero values of w can be represented as a convolution between the projection on $w=0$ plane and the convolution kernel $\tilde{G}(u,v,w)$. Here, $\tilde{G}(u,v,w)$ is the two–dimensional Fourier transform of $G(l,m,w)$, written as:

$$\begin{aligned} \tilde{G}(u,v,w) &= \int\int \mathrm{d}l\mathrm{d}m G(l,m,w) \exp\left[-i2\pi(ul+vm)\right] \\ &\approx \int\int \mathrm{d}l\mathrm{d}m \exp[i\pi w(l^2+m^2)] \exp\left[-i2\pi(ul+vm)\right] \\ &= \frac{i}{w} \exp\{-i\pi[\frac{u^2+v^2}{w}]\} \end{aligned} \tag{4.19}$$

where the small angle approximation is used following Cornwell et al. (2003, 2005, 2008).

According to Equation 4.18, given an image model, the visibility model is easily obtained as follows when the least-misfit gridding function with $x_0 = 0.25$ is used:

1) Multiply the least-misfit correcting function with the image model.
2) Pad zeros to the outer edge of the image model, to make it double-sized.
3) Apply FFT to the two–dimensional image model so as to obtain visibilities $V(u,v,w=0)$.
4) Convolve $\tilde{G}(u,v,w)$ with $V(u,v,w=0)$ to obtain the predicted visibility model $V(u,v,w)$.

[c]We do not use here the three–dimensional coordinate system introduced at the beginning of this chapter; instead, n is still in the range of $[0,1]$ so as to keep the present mathematical analysis consistent with Cornwell et al. (2003).

The steps to calculate the FFT dirty image of the wide field of view from the visibility $V(u, v, w)$ can be summarised:

1) Evaluate $V(u, v, w = 0)$ on the $w = 0$ plane from the given visibility data $V(u, v, w)$.
2) Grid the visibilities $V(u, v, w = 0)$ with the least–misfit gridding function, then apply an inverse FFT to the gridded visibilities.
3) Crop the outer half of the FFT results when $x_0 = 0.25$.
4) Multiply the cropped FFT results by the correcting function, to obtain the FFT dirty image.

The sky image then can be reconstructed from the dirty image using the beam calculated at the phase centre, using CLEAN or the Maximum Entropy method (Cornwell et al. 2008).

Notice that the two procedures listed are different from those introduced in Cornwell et al. (2005, 2008) with the use of a tapering function.

4.2.3.2 Tapering function and criticism

Cornwell et al. (2003, 2005, 2008) proposed that an image-plane tapering function $T(l, m)$ be applied so as to suppress aliasing effects in the image plane. The Fourier transform of

$$G_T(l, m, w) = T(l, m)G(l, m, w) \tag{4.20}$$

then can be written as a convolution between $\tilde{G}(u, v, w)$ and $\tilde{T}(u, v)$[d]

$$\mathscr{F}_{2D}\{G_T(l, m, w)\} = \tilde{G}(u, v, w) * \tilde{T}(u, v)$$

Hence, $\tilde{T}(u, v)$ will be used as a gridding function on the visibility plane. The corresponding correcting function should also be applied. The rationale behind the tapering function remains questionable, however.

The term $T(l, m)$ is chosen as the spheroidal function with a support of $W = 9$ (Cornwell et al. 2003, 2005, 2008). Since the truncated spheroidal function with $\alpha = 0$ is equal to its own Fourier transform (Schwab 1984; Briggs et al. 1999), $\tilde{T}(u, v)$ will help to suppress aliasing effects, as claimed. But this is not a basic justification for the tapering function, since a gridding function will be used anyway for the FFT operation. As our least-misfit gridding function is much better at suppressing aliasing than the spheroidal function (see Chapter 3), it is preferable to the spheroidal function. We introduce our least-misfit gridding function only on the visibility plane, rather than on the image plane, as $T(l, m)$.

According to the degridding steps introduced in Cornwell et al. (2005, 2008), the image model should be multiplied by the tapering function. This appears wrong, as the image model

[d]The Fourier transform of the tapering function $T(l, m)$.

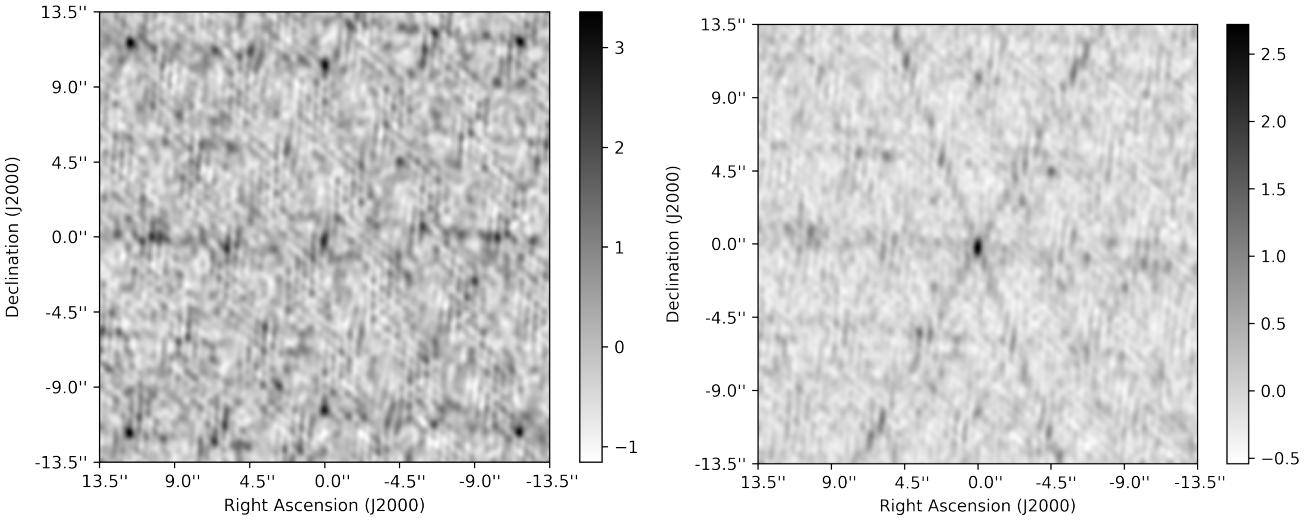


Figure 4.10: Left: FFT dirty image made from degridded visibility data obtained from our proposed degridding procedure. Right: FFT dirty image made from degridded visibility data obtained from the degridding steps suggested by Cornwell et al. (2005, 2008). The same 34-source image model is used in both cases.

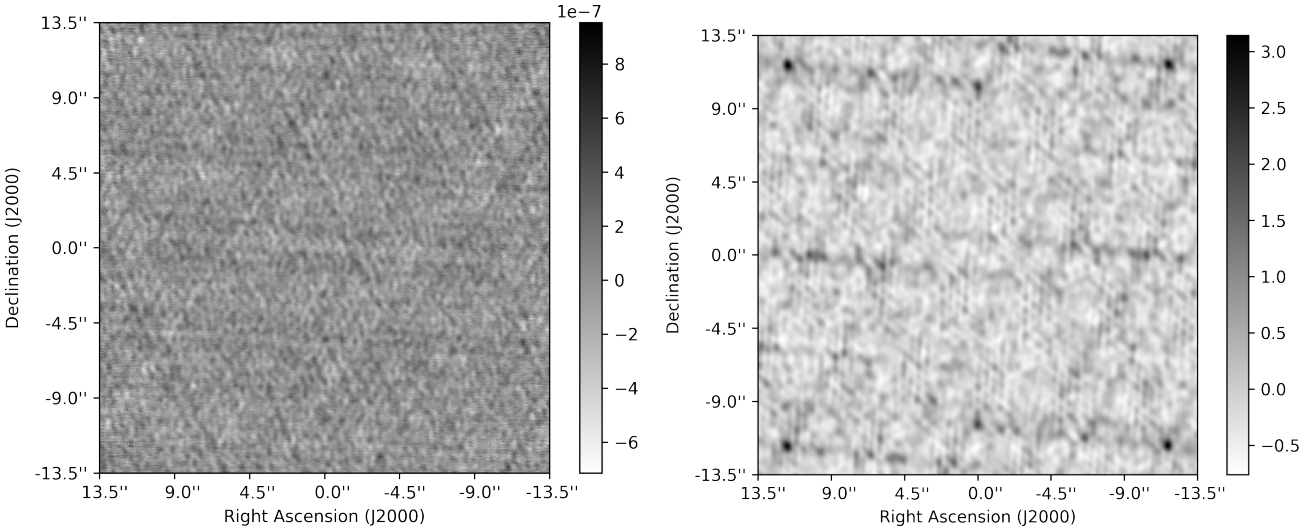


Figure 4.11: Difference maps between the FFT dirty images shown in Figure 4.11 and the DFT image constructed from visibility data obtained via the DFT route.

should be multiplied by the correcting function. The correcting function is calculated either as the reciprocal of the Fourier transform of the chosen gridding function (see Equation 3.20), or is calculated as following Equation 3.18 for the least-misfit gridding function.

Figures 4.10 and 4.11 show the results of a numerical experiment to test the correctness of the degridding steps written in Cornwell et al. (2005, 2008). From the 34–source image model,

we first use our proposed degridding steps to generate degridded visibility data. Then we make a FFT dirty image from it, which is displayed as the left map in Figure 4.10. The baselines are taken from the 34–source VLA A-array snapshot used in Chapter 3. The image size is 900 by 900, with a cell size of 0.03 arcsec. The least-misfit gridding function is used, with $W = 7$ and $x_0 = 0.25$.

As a comparison, we use the degridding steps proposed by Cornwell et al. (2008):

- Multiply the image model by the least-misfit gridding function, which is taken as the tapering function T.
- Pad zeros to the outer edge of the tapered image model, to make it double-sized.[e]
- Perform the two-dimensional FFT.
- Correlate the FFT results with the gridding function to obtain the visibility model.

After the visibility model has been obtained, we make a FFT dirty image, which is displayed on the right in Figure 4.10. Only the central part of the image is well reconstructed, and the sensitivity falls steeply towards the edges. This is because the tapering function gives more weight to pixels at the centre of the image, but less weight to pixels near the edge.

From the same 34–source image model, a DFT visibility model can be constructed via a direct Fourier transform. A DFT dirty image can then be made from this. In Figure 4.11 we show maps of the difference between this DFT dirty image and each of the FFT dirty images. The difference map on the left is at a magnitude of 10^{-7}. The four corner sources are clearly visible on the difference map displayed at right, but not at left. We conclude that the step in which the tapering function enters by multiplication during the degridding steps gives rise to incorrect degridding results. We also believe that no extra tapering function on the image plane is needed.

We know of no data analysis which has actually used the degridding steps proposed in Cornwell et al. (2005, 2008). Our own proposed degridding steps are consistent with the proposal of Venkata (2010).

4.2.4 W–Stacking

Instead of projection, there is another way to deal with the w–term, known as W–Stacking (Humphreys & Cornwell 2011; Offringa et al. 2014). Beginning again from Equation 1.1,[f] the right–hand side can be separated into one part without any w–term and the rest which contains

[e] There is no such step in Cornwell et al. (2005, 2008), but zero-padding is necessary as $x_0 = 0.25$.

[f] We do not use here the three–dimensional coordinate system introduced at the beginning of this chapter; instead, n is still in the range $[0, 1]$ so that the analysis remains consistent with that of Humphreys & Cornwell (2011); Offringa et al. (2014).

the w–term. For each given non–zero w_i, we may rewrite

$$V(u,v,w_i) = \int\int \mathrm{d}l\mathrm{d}m \frac{I(l,m)}{\sqrt{1-l^2-m^2}} \exp\left\{-i2\pi\left[w_i\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\} \exp\left[-i2\pi(ul+vm)\right] \tag{4.21}$$

For each given non–zero w_i, Equation 4.21 is essentially a two–dimensional Fourier transform. By inverting the transform, we have:

$$\frac{I(l,m)}{\sqrt{1-l^2-m^2}} = \exp\left\{i2\pi\left[w_i\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\} \int\int \mathrm{d}u\mathrm{d}v V(u,v,w_i) \exp\left[i2\pi(ul+vm)\right] \tag{4.22}$$

If we integrate both sides along the w axis from the minimum $w_{\min}$ to the maximum $w_{\max}$ on the right–hand side, the result is

$$\frac{I(l,m)(w_{\max}-w_{\min})}{\sqrt{1-l^2-m^2}} = \int_{w_{\min}}^{w_{\max}} \mathrm{d}w \exp\left\{\mathrm{i}2\pi\left[w\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\} \int\int \mathrm{d}u\mathrm{d}v V(u,v,w) \exp\left[i2\pi(ul+vm)\right] \tag{4.23}$$

The theoretical steps giving the dirty image can be summarised as follows:

1) For each w value,
   a) the visibilities with the same w value are gridded and an inverse FFT is applied, followed by an image cropping and a multiplication with the correcting function on the image plane.
   b) a phase shift is then performed on the image product.
2) Sum the results for all w values and then rescale it by multiplying $\frac{\sqrt{1-l^2-m^2}}{(w_{\max}-w_{\min})}$.

In practice, the values of w are discretised into N_w uniform samples along the w axis, Equation 4.23 can be written as:

$$\frac{I(l,m)(w_{\max}-w_{\min})}{\sqrt{1-l^2-m^2}} = \sum_{n=0}^{N_w-1} \exp\left\{i2\pi\left[w_n\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\} \int\int \mathrm{d}u\mathrm{d}v V(u,v,w_n) \exp\left[i2\pi(ul+vm)\right] \tag{4.24}$$

Following Offringa et al. (2014), the separation of two subsequent w values δw should satisfy the criterion:

$$|\delta w 2\pi(\sqrt{1-l^2-m^2}-1)| \ll 1 \tag{4.25}$$

If δw is larger than this, more w samples are needed. Hence, N_w is determined by:

$$N_w \geq 2\pi(w_{\max}-w_{\min}) \max_{l,m}(1-\sqrt{1-l^2-m^2}) \tag{4.26}$$

The choice of N_w has a great impact on the performance of W–Stacking (Offringa et al. 2014).

After N_w has been determined for a given observation, a total of N_w uniformly sampled w–planes will be in play. Each item of visibility data then will be assigned to its nearest w–plane

(Offringa et al. 2014). From Figure 4.12, there are N_w planes on the w axis, ranging from w_0 to w_{N_w-1}. For a particular visibility value $V(u_i, v_i, w_i)$, if w_i is rounded to w_0, this visibility is to be placed on the $w = w_0$ plane. After all visibilities have been assigned to the corresponding w–planes, the visibilities on each of these w–planes are to be gridded.

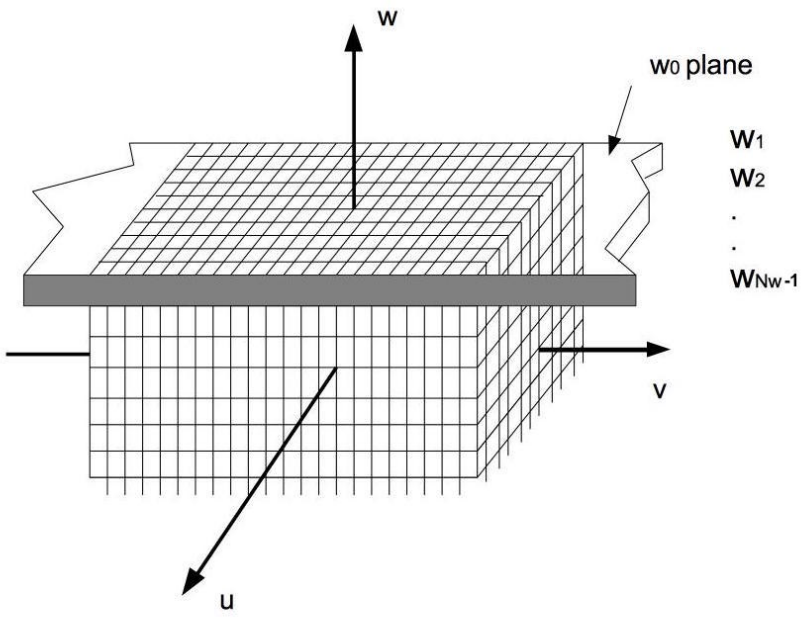


Figure 4.12: There are N_w planes on the w axis, ranging from w_0 to w_{N_w-1}. For a particular visibility value $V(u_i, v_i, w_i)$, if w_i is rounded to w_0, this visibility is placed onto the $w = w_0$ plane. After all visibilities are assigned to w–planes, the visibilities on each w–planes are gridded as shown.

In a further schematic diagram, 4.13, the crosses on the w_0 plane represent visibilities on this plane, and these will be gridded in order to assign values to nearby uv grids. To the gridded visibilities on the w_0 plane will then be applied an inverse FFT and a phase shift, to generate the image produced by all visibilities assigned onto the $w = w_0$ plane.

This process is repeated N_w times on all w–planes, and the right-hand side of Equation 4.24 can be represented as a sum of the N_w image results. Hence, the desired sky image $I(l, m)$ can be reconstructed as:

$$I(l,m) = \frac{\sqrt{1-l^2-m^2}}{(w_{\max}-w_{\min})} \sum_{n=0}^{N_w-1} \exp\left\{i2\pi\left[w_n\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\} \mathscr{F}_{2D}^{-1}\{V(u,v,w_n)\} \quad (4.27)$$

where $\mathscr{F}_{2D}^{-1}$ denotes the two–dimensional inverse Fourier transform operation.

The W–Stacking algorithm can be summarised as follows:

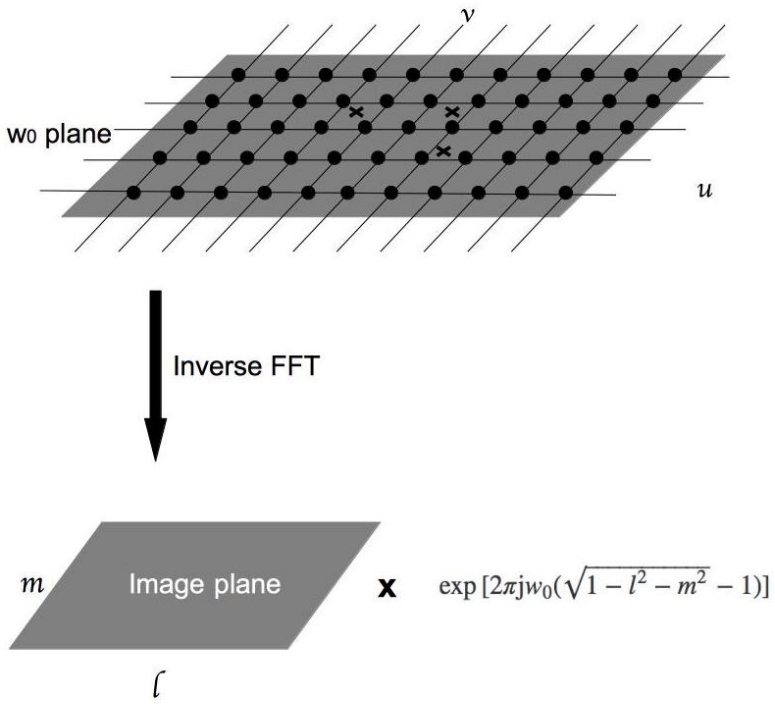


Figure 4.13: The crosses on the w_0 plane represent visibilities on this plane, and they are gridded so as to assign values to nearby uv grids. To the gridded visibilities on the w_0 plane is then applied an inverse FFT and a phase shift, so as to give the image produced by all visibilities assigned onto the $w = w_0$ plane.

1) Determine the number of samples along the w axis N_w; hence the sampled w planes will be distributed uniformly along the w axis from w_0 to $w_{(N_w-1)}$.
2) Assign visibilities to differing w planes according to their w values using the nearest neighbour rule.
3) For each given w–plane $w = w_i$,
   a) the visibilities assigned to it are to be gridded and an inverse FFT applied, followed by multiplication with the correcting function on the image plane.
   b) perform a phase shift to the FFT results by multiplication with $\exp\left\{i2\pi\left[w_i\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\}$.
4) Sum the results for all N_w planes and rescale the image product.

Since the observed visibilities are not fully sampled, the result obtained by the W–Stacking algorithm is the dirty image $I_D(l, m)$, rather than the original sky brightness $I(l, m)$. Hence, a

deconvolution method is required to reconstruct $I(l, m)$ from the dirty image. The degridding process during the deconvolution operation can be summarised as below:

1) Determine the number of w–planes N_w.
2) For each given w–plane $w = w_i$,
   a) Multiply the correcting functions and a phase rotation $\exp\left\{-i2\pi\left[w_i\left(\sqrt{1-l^2-m^2}-1\right)\right]\right\}$ with the image model on the image plane.
   b) Apply FFT to the corrected and phase–shifted image model.
   c) Each image will be correlated with the chosen gridding function so as to assign the gridded visibility values back to the original uv positions.

The degridding results will be N_w sets of visibilities on N_w corresponding w–planes.

We observed that errors have already been introduced by assigning visibilities onto nearest w–planes. In addition, the degridded results introduce further errors during the deconvolution process, because the degridded visibility model will be assigned onto its nearest w–planes instead of where it was originally sampled. A more accurate implementation of W–Stacking is needed.

4.3 Improved W–Stacking

4.3.1 Improved W–Stacking

In this section, we propose an improved W–Stacking method following the suggestion of Dr. Sze Meng Tan (private communication). The mathematical derivation is identical to what has been described in Section 4.2.4, but W–Stacking can be implemented differently in practice to obtain more accurate dirty images with fewer w–planes.

To obtain more accurate dirty images, all visibilities should be gridded into three dimensional grids, rather than projecting every w value onto its nearest w–planes. This means that the w value will be gridded onto grids along the w–axis also; see Figure 4.14 where, on the $w-u$ plane, the visibilities will be gridded into rectangular grids represented by the black points. (The v axis is not plotted, for simplicity.) Hence, instead of a two–dimensional uv plane gridding on each w plane, the visibilities will be gridded onto genuinely three–dimensional grids. Correcting functions will also be applied to all three dimensions on the image plane, so as to cancel out the effects arising from the three–dimensional gridding operation.

In this improved W–Stacking method, the least–misfit gridding function $C(u, v, w)$ proposed in Chapter 3 is used. This is separable:

$$C(u, v, w) = C(u)C(v)C(w)$$

. In practice, we use $C(w)$ first, with a support width W so as to grid all visibilities on to the adjacent W grids along the w–axis. Next, the visibility values assigned on each w–plane are

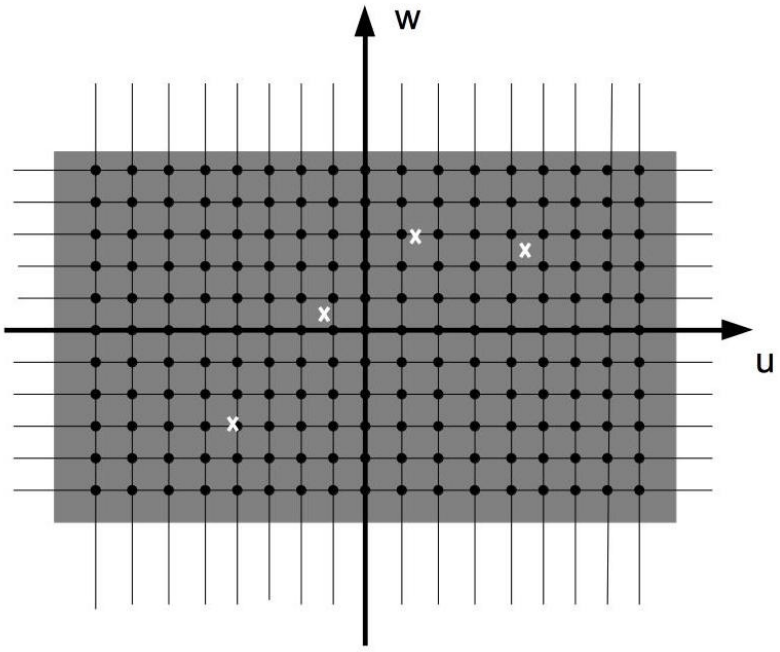


Figure 4.14: On the $w-u$ plane shown, the visibilities are to be gridded into Cartesian grids represented by the black points in the figure. The v axis is not shown, for simplicity. Hence, instead of a two–dimensional uv plane gridding on each w plane, the visibilities will be gridded onto three–dimensional grids.

gridded onto uv grids. Each gridded w–plane is then applied an inverse FFT, an image cropping and a phase rotation. Instead of saving N_w images, we save only the sum of these images, to reduce storage costs.

In view of the differing gridding procedures and the use of the least–misfit gridding function, the number of w–planes is calculated differently. Suppose we are observing an area of sky defined by $l_{\min}, m_{\min}, l_{\max}$ and $m_{\max}$ on a Cartesian grid of $N_x \times N_y$ pixels. To use the least–misfit gridding function, we map the range $[l_{\min}, l_{\min}]$ to $[-x_0, x_0]$. We do the same for m, but for n the range is $n \in (n_{\min}, 0)$, where $n_{\min} = \max_{l,m}(\sqrt{1-l^2-m^2}-1)$. We introduce the normalised three dimensional coordinate shown in Figure 4.15 and based on the coordinate system shown in Figure 3.2. The left coordinate system is centred at $(0,0,0)$. Here $l_{\min}$, $m_{\min}$, $l_{\max}$ and $m_{\max}$ define the FoV, and $n = 0$ represents the tangent plane with $n = \sqrt{1-l^2-m^2}-1$. The normalised three–dimensional coordinate system on the right shows the corresponding normalised field defined by $x = \frac{l}{l_{\max}-l_{\min}} \in [-0.5, 0.5]$ and $y = \frac{m}{m_{\max}-m_{\min}} \in [-0.5, 0.5]$. For simplicity, we write the range of l and m as $l_{\text{range}} = l_{\max} - l_{\min}$ and $m_{\text{range}} = m_{\max} - m_{\min}$

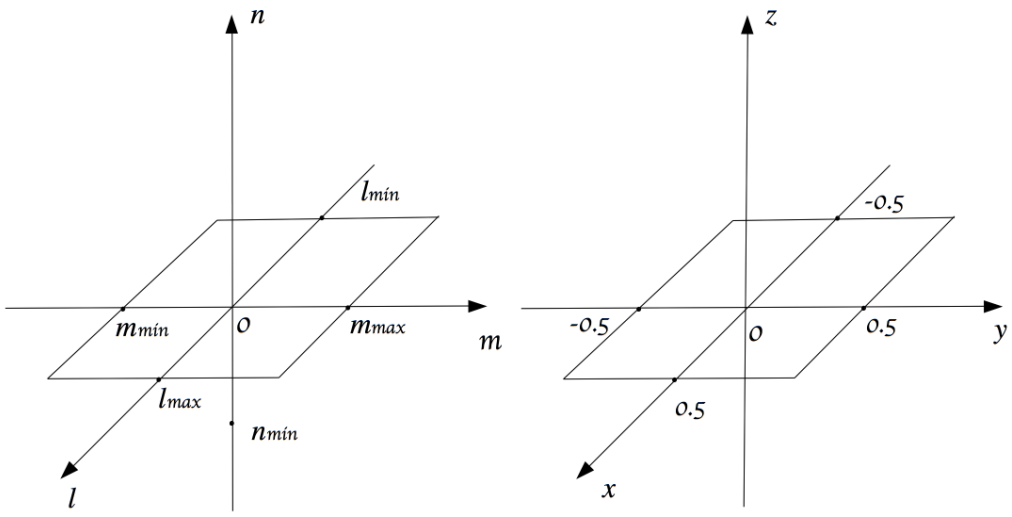


Figure 4.15: The three–dimensional coordinate system formed by the direction cosines (l, m, n) (left) and the normalised coordinate system (x, y, z) (right). The left coordinate system is centred at $(0, 0, 0)$, and $l_{\min}$, $m_{\min}$, $l_{\max}$ and $m_{\max}$ define the FoV, while $n = 0$ represents the tangent plane with $n = \sqrt{1 - l^2 - m^2} - 1$. The normalised three–dimensional coordinate system (right) shows the corresponding normalised field defined by $x = \frac{l}{l_{\max} - l_{\min}} \in [-0.5, 0.5]$ and $y = \frac{m}{m_{\max} - m_{\min}} \in [-0.5, 0.5]$.

respectively.

The map angular size required for the FFT in l is $l_{\text{range}}/(2x_0)$, and is N_x pixels. If u is specified in wavelengths, it is multiplied by $l_{\text{range}}/(2x_0)$ to convert to pixels, so that

$$N_x \geq (u_{\max} - u_{\min}) l_{\text{range}}/x_0 \tag{4.28}$$

We now consider the choice of the w or z grid. We set the phase centre $z = 0$ at $n = 0$ and $z = -x_0$ at $n_{\min}$. In our improved W–Stacking, we do not apply FFT in the z direction, so there is no advantage in oversampling the beam considerably. Consequently,

$$N_z \geq (w_{\max} - w_{\min}) n_{\min}/x_0 \tag{4.29}$$

The number of w–planes using least–misfit gridding function can therefore be determined as:

$$N_{w'} \equiv N_z \geq \frac{\max_{l,m}(1 - \sqrt{1 - l^2 - m^2})(w_{\max} - w_{\min})}{x_0} + W \tag{4.30}$$

The additional W planes enable visibilities located close to the top or bottom w–planes to be gridded to grids on both sides.

The number of w–planes, N_w, determined by Equation 4.26 is more than 1.5 times greater than the first part of $N_{w'}$ determined by Equation 4.30. Thus, for given data, the improved

W–Stacking method may require fewer w–planes than the original W–Stacking method. This point will be discussed further in Section 4.6. $x_0 = 0.25, W \geq 7$ is recommended so as to achieve the single precision limit in the image misfit level.

The corresponding correcting function $h(l, m, n)$ should be adopted, and can be separated as

$$h(l, m, n) = h(l)h(m)h(n)$$

. In Chapter 3, a two–dimensional correcting function $h(l, m)$ was calculated. The correcting function $h(n)$ can be calculated based on $n = \sqrt{1 - l^2 - m^2} - 1$ and a pre–calculated $h(l)$ through quintic interpolation. This interpolation proves to be good even when double precision is used. Hence, $h(n)$ is actually a two–dimensional function of l and m. Instead of applying the correcting function to each w–plane image, it can be applied to the image sum with a consequent saving in computational cost.

In summary, the procedures for our improved W–Stacking method are:

1) Determine the number of samples on the w axis $N_{w'}$; hence the sampled w planes will be uniformly distributed across the w axis from w_0 to $w_{(N_{w'}-1)}$.
2) Convolve the gridding function $C(w)$ with $V(u, v, w)$ in order to grid the visibilities onto the w–planes.
3) Set an all-zero $N_x \times N_y$ matrix $\mathbf{I}$. For each w–plane at $w = w_i$,
   a) Grid the visibilities assigned on this w–plane onto uv grids, followed by a two–dimensional inverse FFT.
   b) Crop the outer half of the FFT image result.
   c) Calculate $\exp\left[i2\pi w_i\left(\sqrt{1 - l^2 - m^2} - 1\right)\right]$, and multiply it with the cropped FFT result.
   d) Add the result to the matrix $\mathbf{I}$.
4) Multiply the correcting function with $\mathbf{I}$, to obtain the final image product.

The corresponding degridding process is:

1) Determine the number of w–planes, $N_{w'}$.
2) Multiply the correcting function by the two–dimensional image model.
3) For each w–plane at $w = w_i$,
   a) Apply a phase rotation $\exp\left\{-i2\pi\left[w_i\left(\sqrt{1 - l^2 - m^2} - 1\right)\right]\right\}$ to the corrected image model.
   b) Pad zeros around the phase–shifted image model to make it double–sized, and follow this with a two–dimensional FFT.
4) Having obtained $N_{w'}$ sets of two–dimensional FFT results, a three–dimensional correlation is performed in all three directions using the least–misfit gridding function.

From the degridding process, the improved W–Stacking method allows us to put degridded visibilities back to the positions where the observed visibilities were originally sampled. In the original W–Stacking method, in contrast, the degridding results are specified on uniformly sampled w–planes. Degridding errors are reduced by using the improved W–Stacking method.

The original W–Stacking method and the improved W–Stacking method will be compared, using simulated data, in Section 4.5.

4.3.2 Computational cost of the improved W–Stacking method

We now calculate the computational and storage cost for the improved W–Stacking method. Following the computational cost calculated in 3.5.2, there are $2N_vW^3$ operations during the three–dimensional gridding, where N_v is the number of visibilities. For the inverse FFT, each w–plane involves $N_xN_y/(4x_0^2)\log[N_xN_y/(4x_0^2)]$ operations. The phase rotation requires $N_xN_yN_{w'}$ operations. After that, the sum of $N_{w'}$ images requires $N_xN_yN_{w'}$ operations. Finally, two multiplications[g] with the correcting function are applied, involving $2N_xN_y$ operations. These operations add up to $2N_vW^3 + \left[\frac{N_{w'}}{4x_0^2}\log\left(\frac{N_xN_y}{4x_0^2}\right) + 2N_{w'} + 2\right]N_xN_y$.

The program for our improved W–Stacking method can be modified to reduce the memory storage required. For a certain w–plane, the image result after going through gridding, FFT and phase rotation will be added to the saved image matrix $\boldsymbol{I}$. The storage used for this image result can then be freed up. Hence, instead of saving all $N_{w'}$ planes, only one $N_x \times N_y$ matrix need be saved until all of the w–plane images have been added up. The matrix will then be corrected only once. For single precision operations in our program, the computational and storage cost for the improved W–Stacking are summarised in Table 4.1.

Table 4.1: Computational and storage cost of the improved W–Stacking method

Process	Computational cost	Storage cost
Observe		$8N_v$
Three–dimensional gridding	$2N_vW^3$	$2N_xN_y/(x_0^2)$
Inverse FFT	$N_{w'}N_xN_y/(4x_0^2)\log[N_xN_y/(4x_0^2)]$	$8N_xN_y$
Phase rotation	$N_xN_yN_{w'}$	$4N_xN_y$
Sum	$N_xN_yN_{w'}$	$4N_xN_y$
Correction	$2N_xN_y$	$4N_xN_y$

A factor 8 represents the 8 bytes needed to store both the real and complex part of an item of visibility data. The gridding process produces a complex $N_x/2x_0$ by $N_y/2x_0$ matrix, rather than $N_x \times N_y$. The complex FFT also requires 8 bytes for each pixel, and only the cropped FFT

[g]The image sum will be multiplied with $h(l,m)$ and $h(n)$.

result will be used and added. After phase rotation, however, only the real part of the results need to be saved.

We tested the 34-source wide–field data on our computer with 6-core Intel i7-4930 processor and 12GB memory; the gridding time is about $15ns \times N_v W^3$, and it effectively took $327ns$ to compute the FFT value on each pixel with $x_0 = 0.25$ for each plane. The DFT time is about $36ns$.

In terms of the computational cost, the difference between the original and improved W–Stacking methods lies in the process of gridding and the w–plane numbers. In original W–Stacking, the computational cost for two dimensional gridding is $2N_v W^2$.

4.4 N–Faceting

We now propose a method which also involves multiple n-planes, like the three–dimensional Fourier transform method, but in which no three–dimensional Fourier transform is needed. We call this method 'N–Faceting'. The proposed least–misfit gridding function will be implemented so as to improve the image plane accuracy.

For any wide-field observation, the celestial sphere within the field of view can be divided into several n–planes, or n–facets, as shown in the schematic diagram in Figure 4.16. We see that there are 6 planes on the n axis, and the top plane is the $n = 0$ plane or tangent plane.

We work from Equation 4.2, and for a given n–facet at $n = n_i$, we have

$$V(u, v, w) \exp(i2\pi w n_i) = \int \int \frac{\mathrm{d}l\mathrm{d}m}{n} I(l, m) \exp[-i2\pi(ul + vm)] \tag{4.31}$$

In this way, the brightness on the n–facet can be calculated via a two–dimensional inverse Fourier transform from the phase–shifted visibilities. By summing all n–facet images and making a projection onto the tangent plane, we obtain the final two-dimensional celestial sphere image. Mathematically speaking, N-Faceting is equivalent to the three-dimensional Fourier transform method.

Because of the incomplete sampling of visibilities, the result of the inverse Fourier transform on each n–facet is a dirty image. The local dirty beam is readily obtained via the inverse Fourier transform of the phase–rotated sampling function on this facet. The deconvolution process on each plane has to be conducted using the local beam, and so a three–dimensional deconvolution is required.

These n–facets can be saved in a three–dimensional matrix, termed here a 'cube', which can easily be deconvolved by CLEAN or MaxEnt with a three–dimensional dirty beam. This dirty beam is also a cube comprising dirty beams calculated on every individual n–facet; this is different from what we defined in Equation 4.11. We implement a three–dimensional Högbom

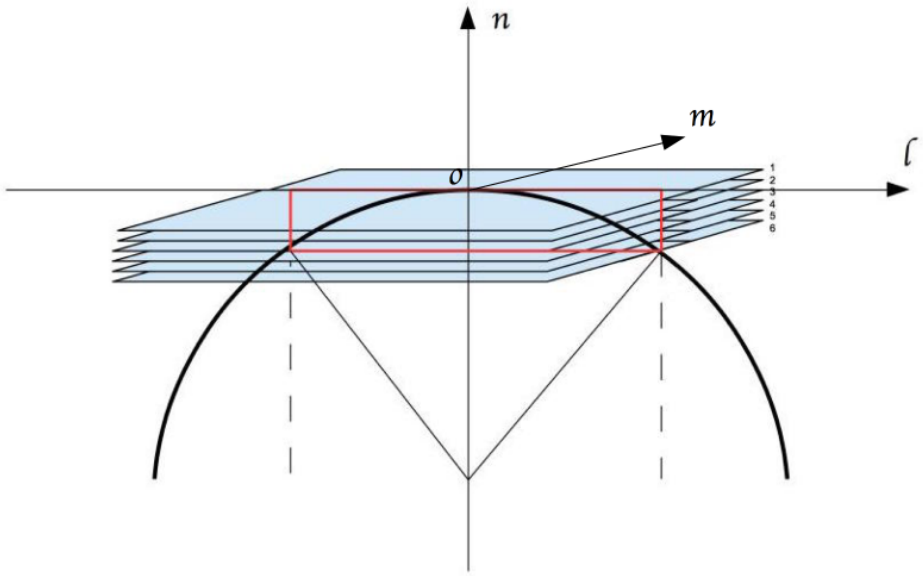


Figure 4.16: The celestial sphere within the field of view has been divided into 6 planes along the n axis. The top plane is the $n = 0$ plane or tangent plane.

CLEAN. In each iteration, a scaled three–dimensional dirty beam is subtracted from the dirty image cube or the three–dimensional residuals at its brightest position. The position and flux of the CLEAN component is recorded and added to a three–dimensional image model, starting from an all–zero three–dimensional matrix. The iteration stops when the brightest pixel on the residual cube is below a chosen threshold, or when the user instructs. After that, we can project the 'CLEANed' cube onto the tangent plane and then make a convolution with a two–dimensional restoring beam. In practice, we display the projected reconstruction as a 2-D image every (say) 100 cycles of CLEAN, and then convolve with a 2-D restoring beam to monitor the process. The convolution between the three–dimensional model and the three–dimensional dirty beam can be stored in order to calculate χ^2.

No residuals will be added to the final image product. The dirty image on every n–plane should have the same noise level, since it is constructed from the same visibility data with different phase rotation applied. We cannot therefore add the sum of the residuals, as this would simply add the sum of the noises to the final image. Also, we cannot determine the choice of residual image if only one residual is to be added. It seems more reasonable to add the averaged residual image. This situation leads us to consider whether the operation of adding the residuals to the reconstructed image is necessary.

We use a different method to determine the number of n–facets N_n rather than N_{planes},

determined via Equation 4.12. We revisit the Bayesian resolving power formula in Equation 2.20

$$(\Delta x_{ij})^{-1} = \partial^2_{ij} B(\mathbf{x})_{|\mathbf{x}=\mathbf{0}} = 4\pi^2\sigma^2 \sum_k \frac{(u_k)_i (u_k)_j}{\sigma_k^2}$$

, where we can obtain the separation between two adjacent n–facets as

$$\Delta n \leq \sqrt{\Delta x_{22}} \tag{4.32}$$

By use of the Bayesian resolving power formula, we may ensure that n–axis is reasonably sampled.

Therefore, the number of n–planes is determined by

$$N_n \geq \frac{n_{\max} - n_{\min}}{\sqrt{\Delta x_{22}}} \tag{4.33}$$

We usually take N_n to be a multiple of a few times the right–hand side, for better sampling on the n–axis.

In summary, the steps involved in N–Faceting for wide field imaging are:

1) Determine the number of the n facets N_n, then obtain an array of the n values for the N_n planes $\{n_1, n_2, n_3, ..., n_{N_n}\}$.
2) For each given n_i:
   a) For each w_j, shift the phase of the visibilities by a multiplication with exp $(i2\pi n_i w_j)$.
   b) Convolve the two–dimensional gridding function $C(u, v)$ with the phase–shifted visibilities on uv grids.
   c) Apply a two–dimensional inverse FFT to the gridded visibilities, and crop the outer half of the image product.
   d) Multiply with the two–dimensional correcting function $h(l, m)$.
3) Deconvolve the sky brightness distribution using a double–sized three–dimensional dirty beam $B_{3D}(l, m, n)$ from the image cube.
4) Project the sky brightness onto the $n = 0$ plane and convolve with the restoring beam.

The imaging process for each of the n facets can be done in parallel, which will save computing time.

The corresponding degridding process from a three–dimensional image cube model can be summarised as:

1) For each given n_i:
   a) Multiply the two–dimensional correcting function by the n_ith facet of the image model and then pad zeros around it to make the facet double–sized.
   b) Apply a two–dimensional FFT.
   c) Match the FFT results with the gridding function to assign them back to the positions where the original visibilities were sampled.

d) Apply a phase shift by a multiplication with $\exp(-i2\pi n_i w_j)$ for each w_j, to obtain the predicted visibilities.

2) Add the degridded visibilities obtained from each facet together as the final degridding result.

The computational cost involved in N–faceting, from obtaining the visibility data to making N_n different dirty images, is summarised in Table 4.2.

Table 4.2: Computational and storage cost of the N–Faceting method

Process	Computational cost	Storage cost
Observe		$8N_v$
Phase rotation	$N_v N_n$	$8N_n N_v$
Gridding	$2N_n N_v W^2$	$2N_n N_x N_y / x_0^2$
Inverse FFT	$N_n N_x N_y / (4x_0^2) \log[N_x N_y / (4x_0^2)]$	$4N_n N_x N_y$

The computational cost of phase rotation and gridding is very expensive, because all visibilities have to be rotated and gridded N_n times, making the N–Faceting method very slow. Compared to the three–dimensional Fourier transform method, N–Faceting not only removes the gridding and FFT process on the n–axis, but also benefits from use of the position–independent beam, so that in theory it should give a more accurate reconstruction than methods which use only a single tangent plane beam.

4.5 Experiments and results

In this section we present numerical experiments and results for the performance of our proposed improved W–Stacking method and N–Faceting method. We then compare our improved W–Stacking with the W–Projection method implemented in `CASA`, and with the original W–Stacking method implemented in `WSClean`.

4.5.1 Performance of improved W–Stacking

To test the performance of our improved W–Stacking, we use the same data as in Figure 4.1. According to our new formula to determine the number of w–planes, in Equation 4.30, we use 38 w–planes. The dirty image produced via the improved W–Stacking method is shown in the left map in Figure 4.17, while the right map is the difference between the dirty image and the DFT dirty image made on the celestial sphere. We use the least–misfit gridding function with $W = 8$ and $x_0 = 0.25$. The difference map has a RMS value of 1.95×10^{-9} using double precision. The primary beam correction operation is used, and the primary beam is produced

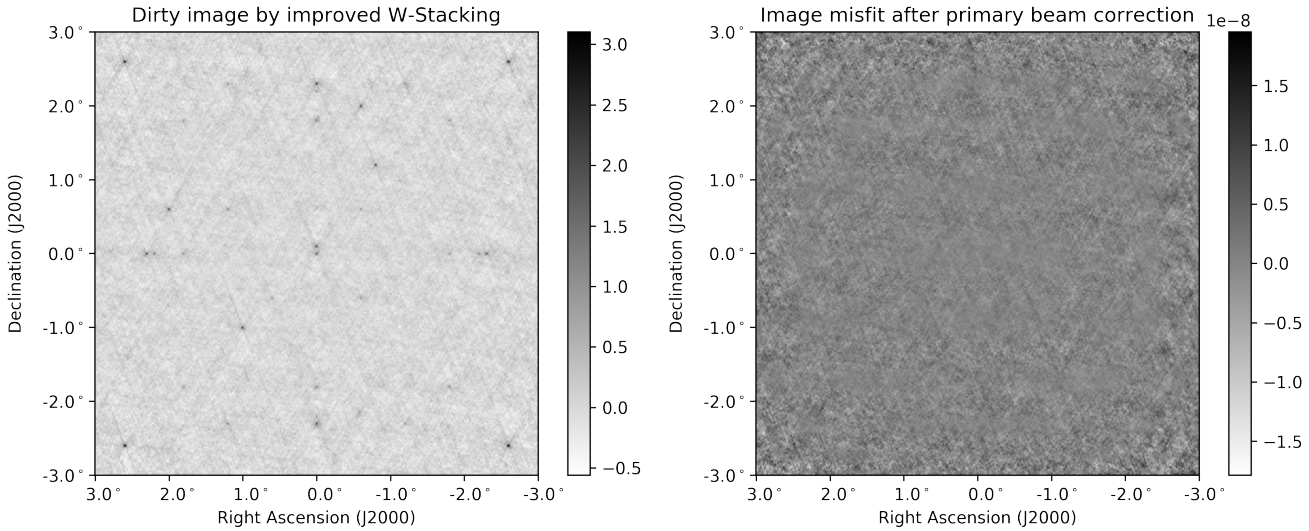


Figure 4.17: Dirty image constructed by the improved W–Stacking method (left), and its difference from the DFT dirty image (right). The least–misfit gridding function is used with $W = 8$ and $x_0 = 0.25$.

by the `clean` command in `CASA`. The fluxes of the corner sources on the DFT dirty map are not exactly 3 Jy, because of low image resolution. The upper right corner source is 2.91 Jy.

To emphasise the importance of the correcting function, we produce the dirty image via the improved W–Stacking method without applying any correcting function; this is shown in the upper left map in Figure 4.18, and the difference between this uncorrected dirty image and the DFT dirty image is shown in the upper right map. In the uncorrected dirty image, sources 2 degrees from the phase centre cannot be seen by eye. The RMS value of the difference map is as large as 7.80×10^{-2}.

If we apply only the correcting function $h(l, m)$ but ignore the correction with respect to n, we obtain the image in the lower left map, and its difference from the DFT dirty image is shown in the bottom right map. Although the four corner sources can be seen in the l, m corrected dirty image, their fluxes are only 1.62 Jy, 44% smaller than their original fluxes, so that further correction taking n into account is needed. The difference map has a RMS value of 2.85×10^{-2}, which is slightly better than when no correcting function is used. The correcting functions $h(l, m)$ and $h(n)$ are shown in Figure 4.19.

We plot the RMS value of the three difference maps against the normalised coordinate in Figure 4.20. The dirty image without any correction has a worse image misfit than the l, m–corrected dirty image. We can see by eye more sources from the partially–corrected dirty image, and it has a better image misfit level than the dirty image without any correction. To obtain the best image misfit and full source information, we should use the three–dimensional correcting

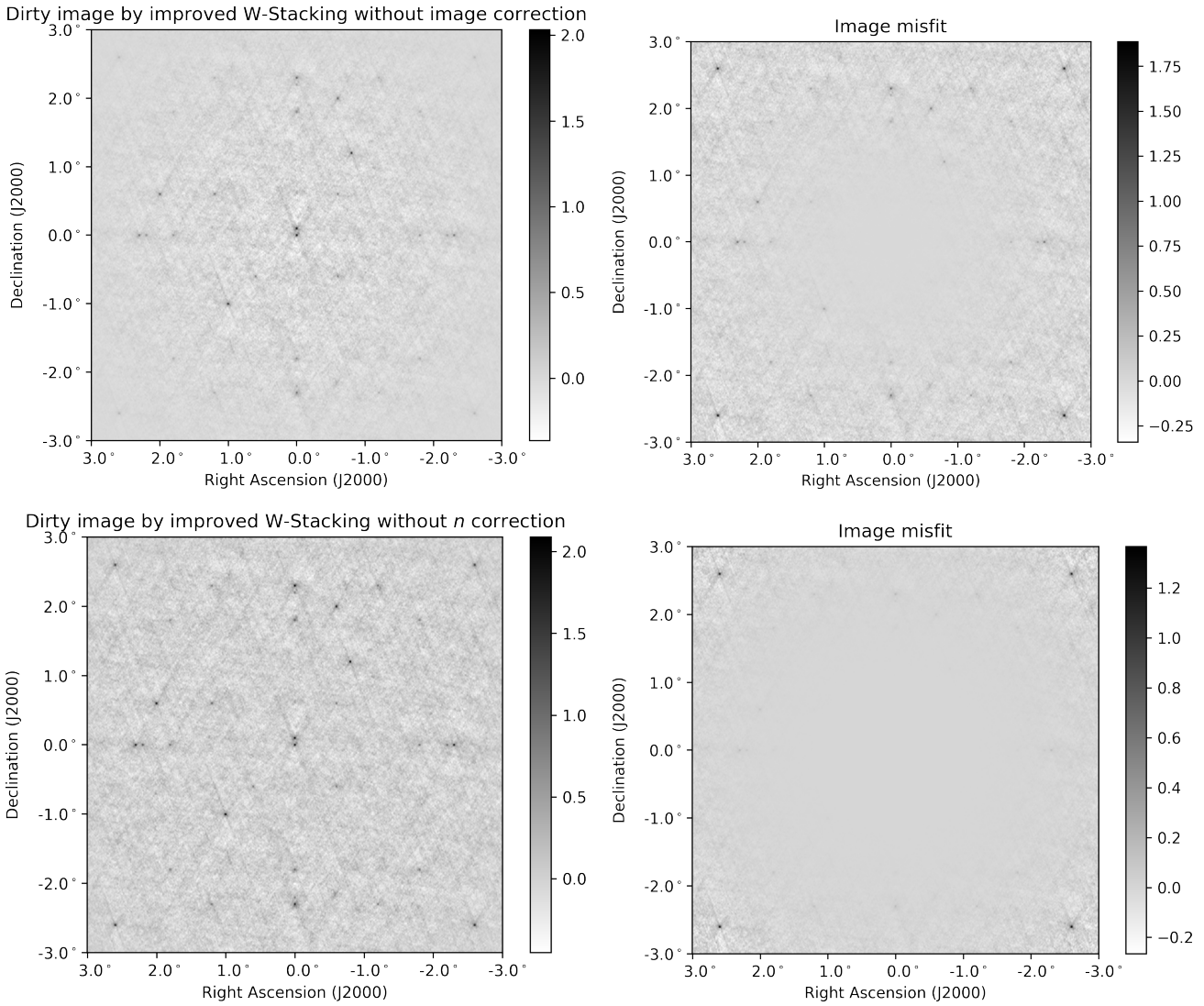


Figure 4.18: The upper two maps demonstrate the dirty image and corresponding image misfit when no image correction has been applied. When only l and m correction is applied, the corrected dirty image and its image misfit from the DFT dirty image are shown in the lower maps.

function, which is based on fully three–dimensional gridding. The original W–Stacking method also needs a correcting function, however, since it uses the nearest neighbour rule on the n–axis along with a two–dimensional gridding process, and its correcting function is therefore different, especially in the n dimension. We have not found information about the correcting function used with the original W–Stacking method, except for a so–called 'windowed low-pass filter' mentioned in `WSClean` manual [h], which is used to attenuate aliased sources.

[h]This can be accessed at https://sourceforge.net/p/WSClean/wiki/WeightingAndGridding/

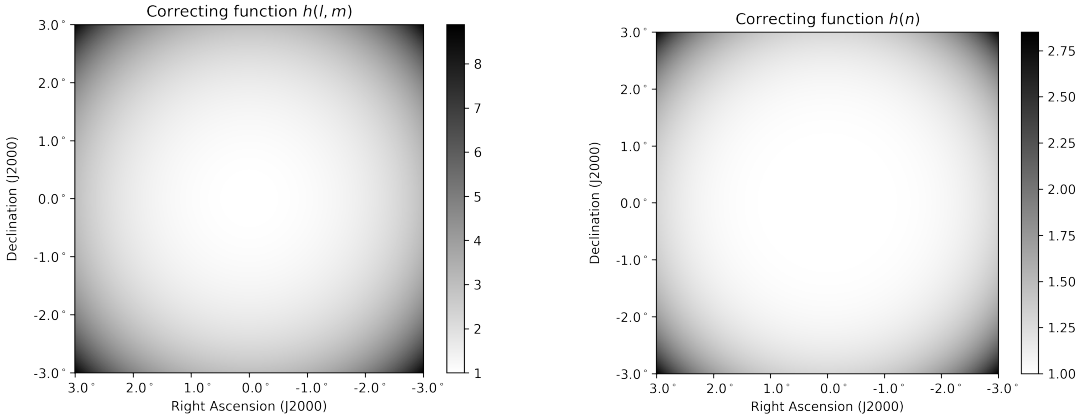


Figure 4.19: The correcting function for the (l, m) plane is shown on the left. The correcting function $h(n)$ is shown at right.

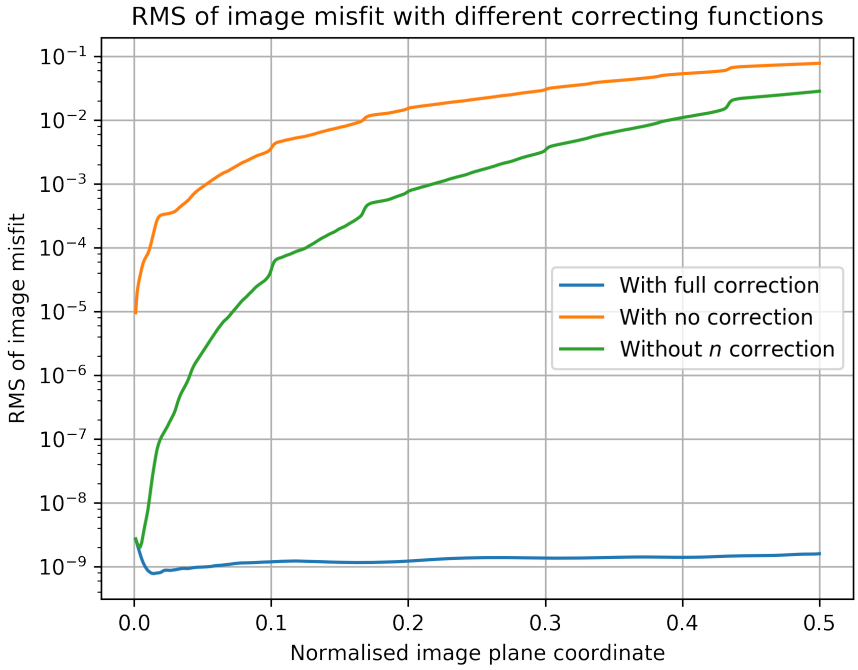


Figure 4.20: RMS results of the three difference maps plotted against the normalised image plane coordinate. The three dirty images are respectively fully corrected, partially corrected and uncorrected.

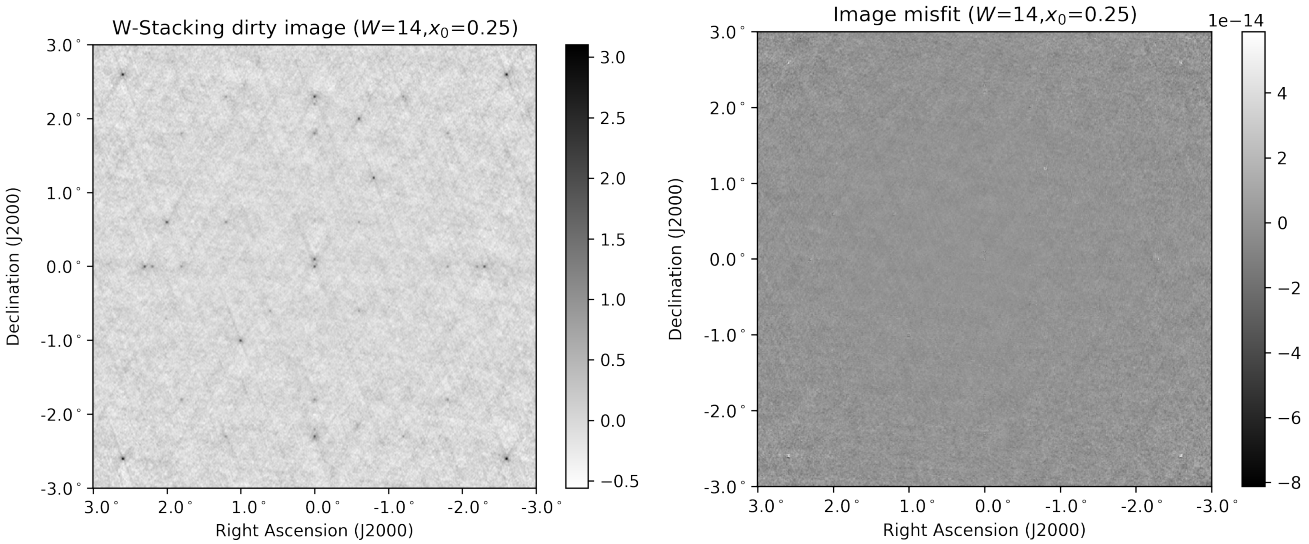


Figure 4.21: Demonstration of the dirty image constructed via the improved W–Stacking method (left), and its difference from the DFT dirty image (right). The least–misfit gridding function is used with $W = 14$ and $x_0 = 0.25$.

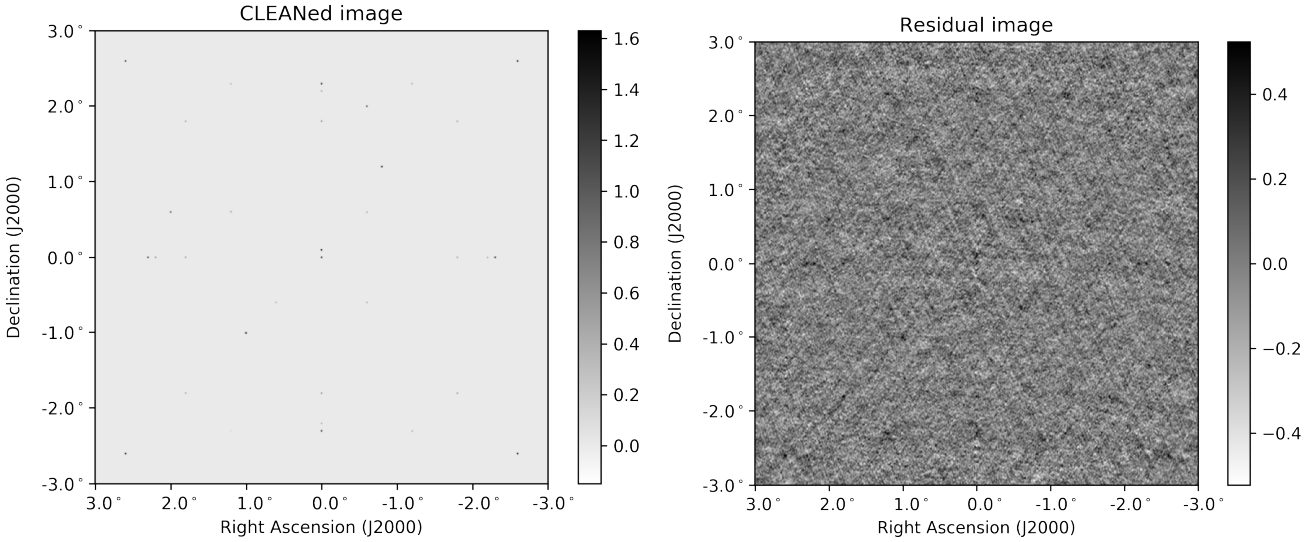


Figure 4.22: Reconstructed image and residual image from a dirty image made via our improved W–Stacking method. The left map shows a reconstructed image by Högbom CLEAN, and the right map is the residual image.

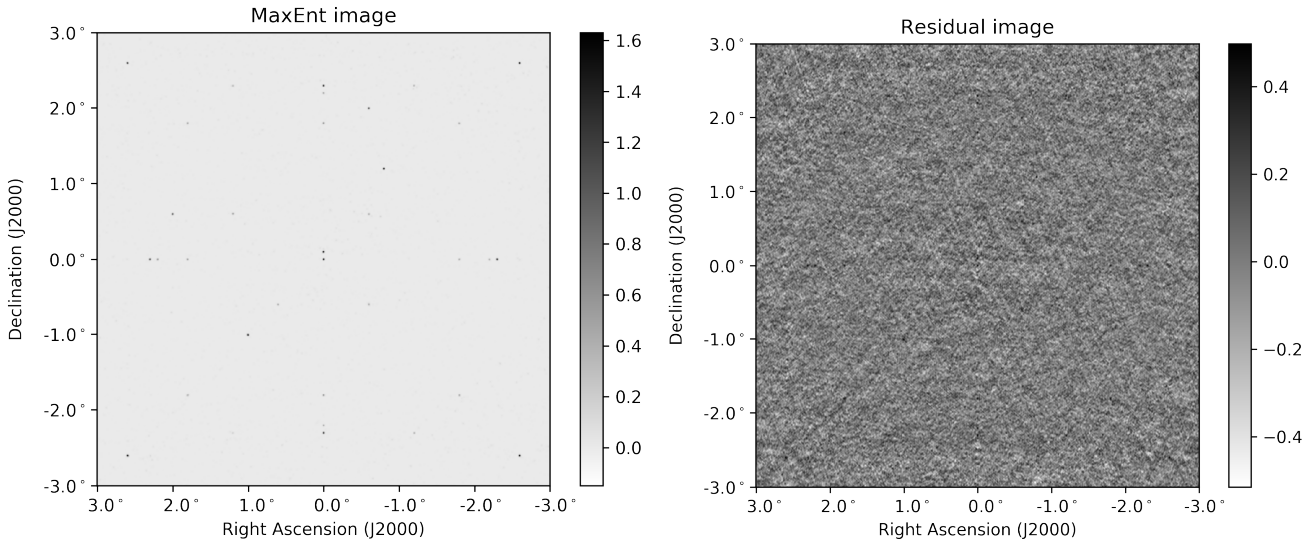


Figure 4.23: Reconstructed image and residual image from a dirty image made via our improved W–Stacking method. The left map shows an image reconstructed by MaxEnt, and the right map is the residual image.

We also demonstrate a dirty image made via the improved W–Stacking method using the least–misfit gridding function with $W = 14$ and $x_0 = 0.25$. The number of w–planes is now 44. The resulting dirty image is shown in the left map in Figure 4.17, and the right map is the difference map between the FFT and DFT dirty images. The difference map has a RMS value of 3.86×10^{-15}, achieving the double precision limit.

We add Gaussian noise of $0.1 Jy/Beam$ to the 34-source data, and then repeat the process above to construct a dirty image via the improved W–Stacking method. The RMS value of the dirty image misfit is still at a magnitude of 10^{-9} with $W = 8$ and $x_0 = 0.25$. We then apply both the Högbom CLEAN and Maximum Entropy method to reconstruct the sky brightness distribution; see Figure 4.22 and Figure 4.23. The left maps are images reconstructed by Högbom CLEAN and MaxEnt (Gull & Skilling 1991), and the right maps are residual images. The images have not been corrected by the primary beam.

4.5.2 Performance of N–Faceting

We work with the same generated 34-source data, but with Gaussian noise added with magnitude $0.1 Jy/Beam$. For N–Faceting, according to Equation 4.33, we make a choice of 32 n–facets, and thereby produce 32 sets of phase–shifted visibilities. We then apply an inverse FFT to each of the gridded visibility dataset, followed by image correcting and cropping. As a result we obtain 32 dirty images on 32 n–facets. We demonstrate 16 dirty images of these in Figure 4.24. If the tangent plane is numbered one, the four dirty images shown in the first row from left to

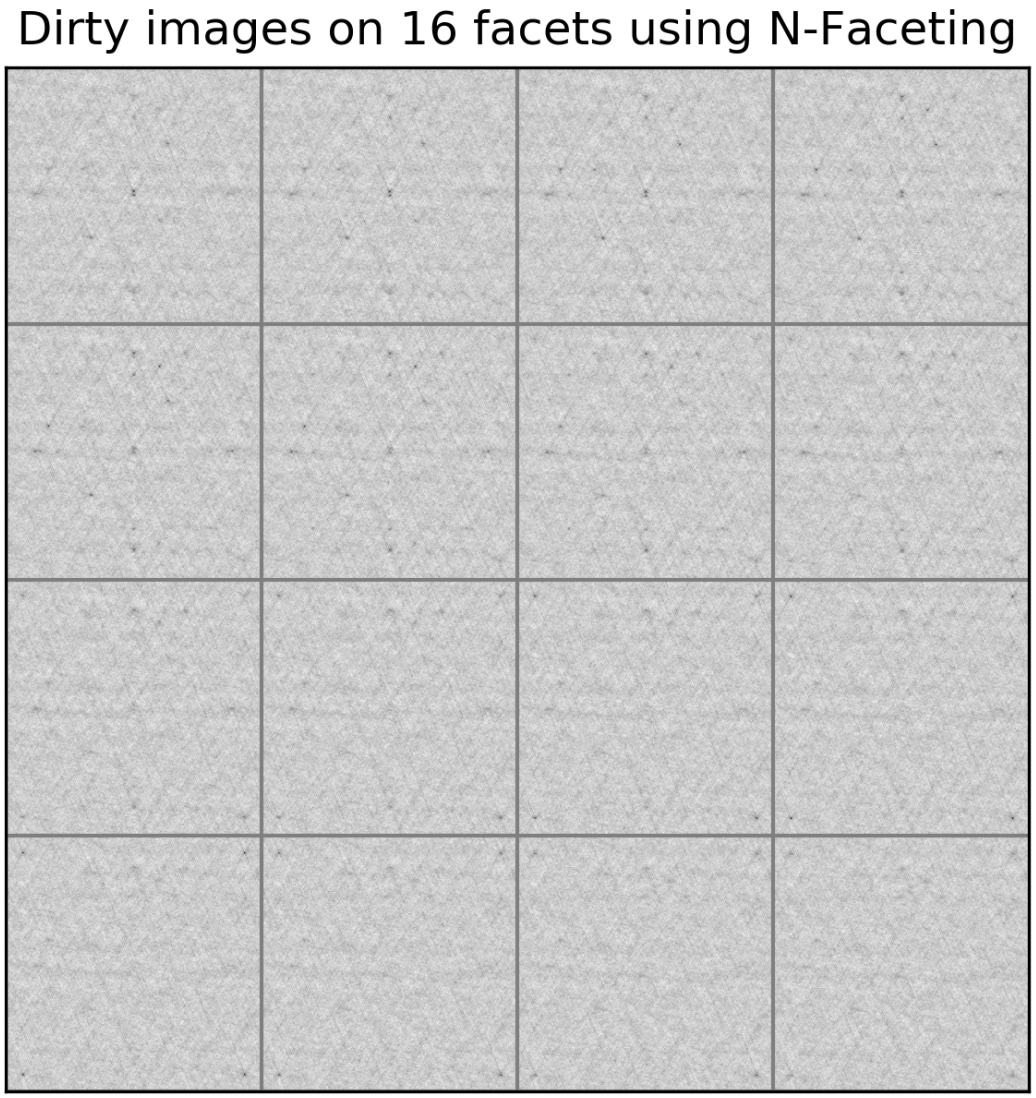


Figure 4.24: Demonstration of 16 dirty images out of 32 facets of dirty images when using the N–Faceting method. The first map on the first row is the tangent plane dirty image. If it is numbered one, the four dirty images shown in the first row from left to right would have plane numbers 1, 3, 5 and 7. The numbers in the second row of images would be 9,11,13 and 15, and so on. In the tangent plane (first facet), we find only two central sources, with other sources blurred or invisible, because these other sources are located on facets below the tangent plane. In the 6th facet, the two central sources are blurred, but sources farther away are clearly visible. The corner sources can be seen clearly only after the 9th facet. At the 31th facet the corner sources fade away, because this facet is below where the four corner sources lie on the celestial sphere.

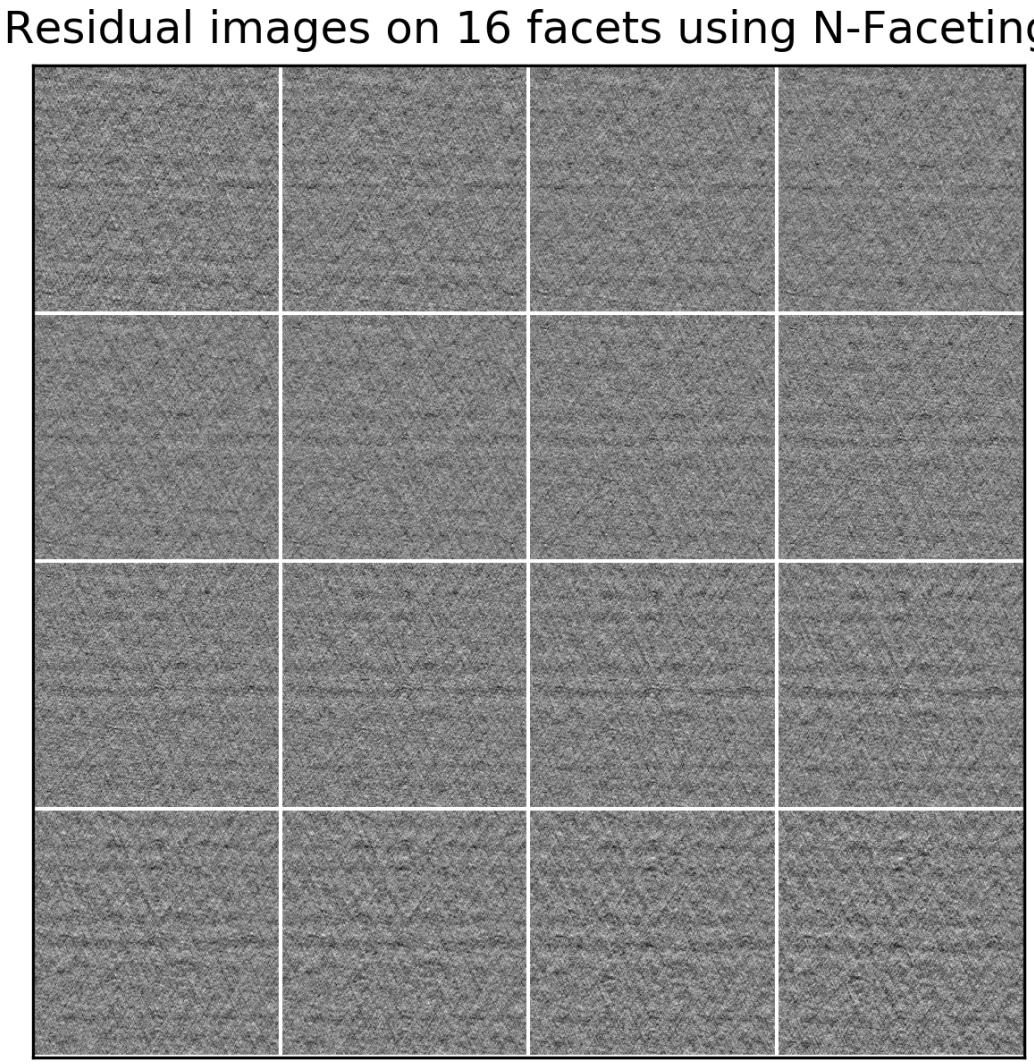


Figure 4.25: Demonstration of 16 residual images out of 32 facets of residual images when using the N–Faceting method. The order of the facets are arranged as in Figure 4.24.

right would have plane numbers 1, 3, 5 and 7. The numbers in the second row of images would be 9,11,13 and 15, and so on. The 32 dirty images make up a dirty image cube, which can be deconvolved using a three–dimensional dirty beam consisting of 32 local dirty beams.

Based on the source locations, we expect to see different sources on different n–facets; sources near the phase centre are expected to be seen on facets close to the tangent plane, and sources far away should be found on facets far below the tangent plane. This trend is demonstrated in Figure 4.24. In the tangent plane, we find only two central sources, with other sources blurred or invisible, because these other sources are located on facets below the tangent plane. In the 6th facet, the two central sources are blurred, but sources farther away are clearly

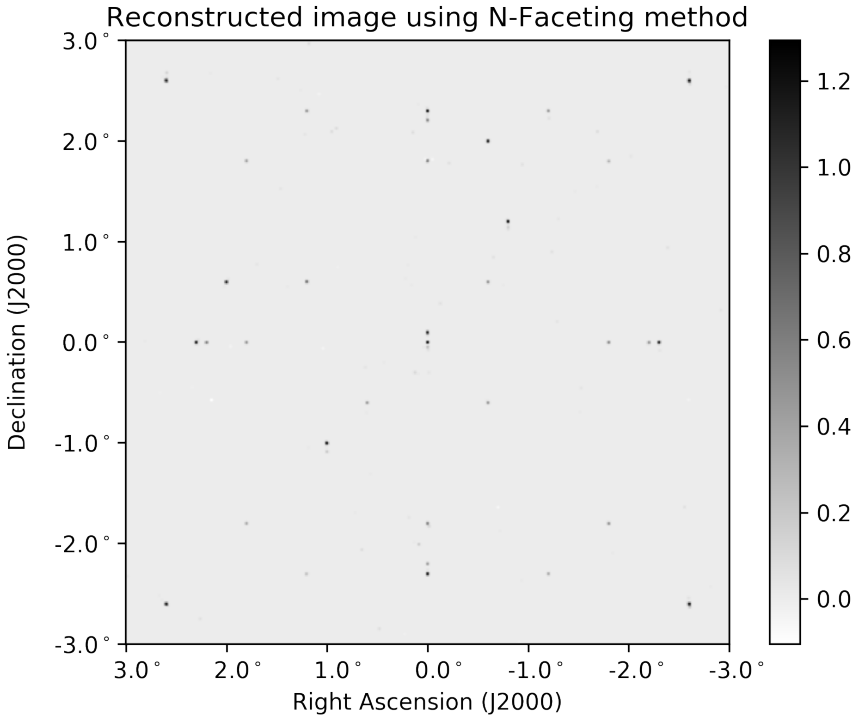


Figure 4.26: The projected image reconstructed via Högbom CLEAN using the N–Faceting method.

visible. The corner sources can be seen clearly only after the 9th facet. At the 31th facet the corner sources fade away, because this facet is below where the four corner sources lie on the celestial sphere.

We use Högbom CLEAN to reconstruct the sky image; the 32 corresponding residual images can be seen in Figure 4.25. The deconvolution result is also a three–dimensional image cube, and we project the cube onto the tangent plane. After convolving it with a two–dimensional restoring beam, we display the final two-dimensional reconstructed image in Figure 4.26. It has not been primary–beam corrected.

It is noted that our N–Faceting method is still novel, future refinements are expected in the future.

4.5.3 Performance of `WSClean` W–Stacking

The original W–Stacking method is implemented in `WSClean`. Here, we conduct numerical experiments to demonstrate the performance of the `WSClean` W–Stacking method. Again we begin by using the same noiseless 34-source data in `WSClean` to construct a dirty image. `WSClean` requires 47 planes, whereas the improved W–Stacking method only requires 38 planes if $W = 8$ is used.

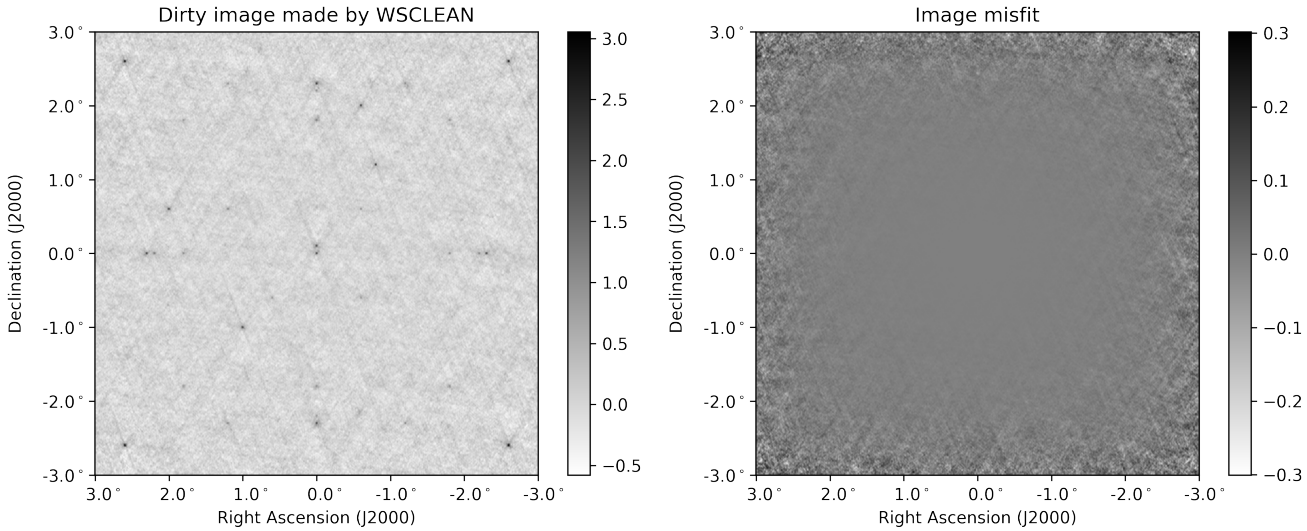


Figure 4.27: Dirty image made through the W–Stacking method implemented in `WSClean` at left, and its difference from the DFT dirty image at right. Primary beam correction has been applied.

We demonstrate the dirty image made by `WSClean` in the left map in Figure 4.27; the right map is the difference between the dirty image and the DFT dirty image. The difference map has a RMS value of 2.34×10^{-2}, much worse than when the improved W–Stacking method is used. Primary beam correction has been applied. The fluxes of the corner sources are 2.72 Jy on the dirty image made using `WSClean`, and are about 6% less than on the DFT dirty image. This occurs also for other three corner sources.

If we make a double–sized dirty image by `WSClean` with the same cell size, but retain only the central half of the image, we obtain the cropped dirty image in the left map in Figure 4.28; the right map is the difference between the cropped dirty image and the DFT dirty image. The difference map has a RMS value of 2.8×10^{-3}, about 10 times better than without image cropping. As a result, the flux of the top right corner source becomes 2.91 Jy on the `WSClean` dirty image, which is satisfactorily accurate. When the required image size is doubled, however, the w–planes `WSClean` needed increase to 186.

Both RMS results are plotted against the normalised coordinate in Figure 4.29. The image misfit levels for both images increase monotonically with the distance from the phase centre. The smallest RMS value of the image misfit is at a magnitude as small as 10^{-5}, but it increases rapidly towards the image edge. As gridding function, we used the default Kaiser–Bessel windowed sinc function with a width of 7. The rapid increase shown is very likely due to the relatively small sampling rate of the gridding function look–up table.

It is evident that `WSClean` is liable to produce inaccurate dirty images, especially underes-

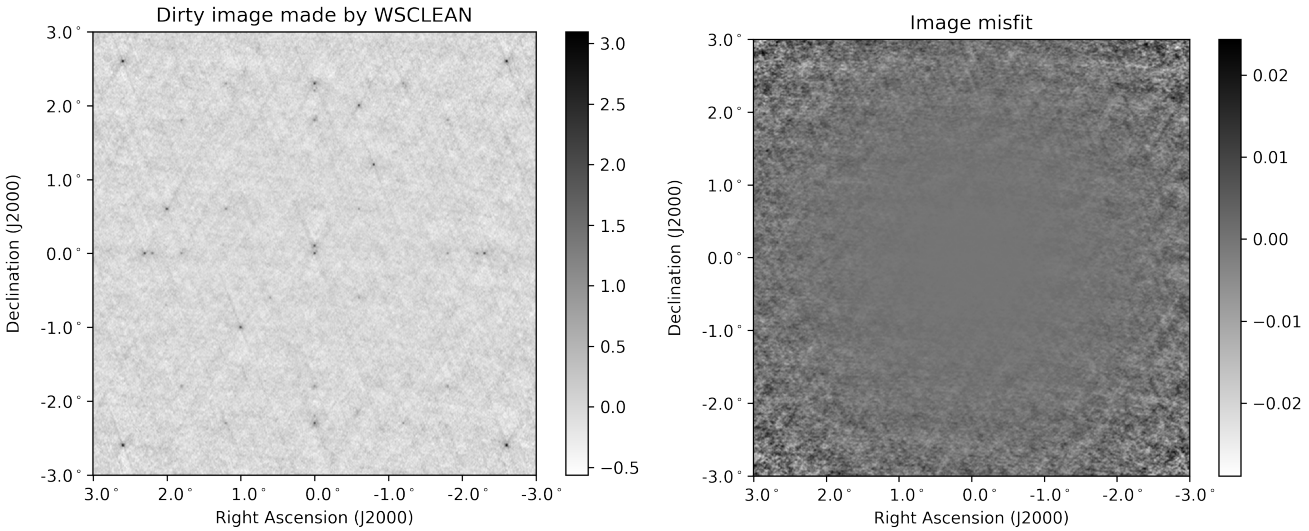


Figure 4.28: The cropped double–sized dirty image made through the W–Stacking method implemented in `WSClean` at left, and its difference from the DFT dirty image at right. Primary beam correction has been applied.

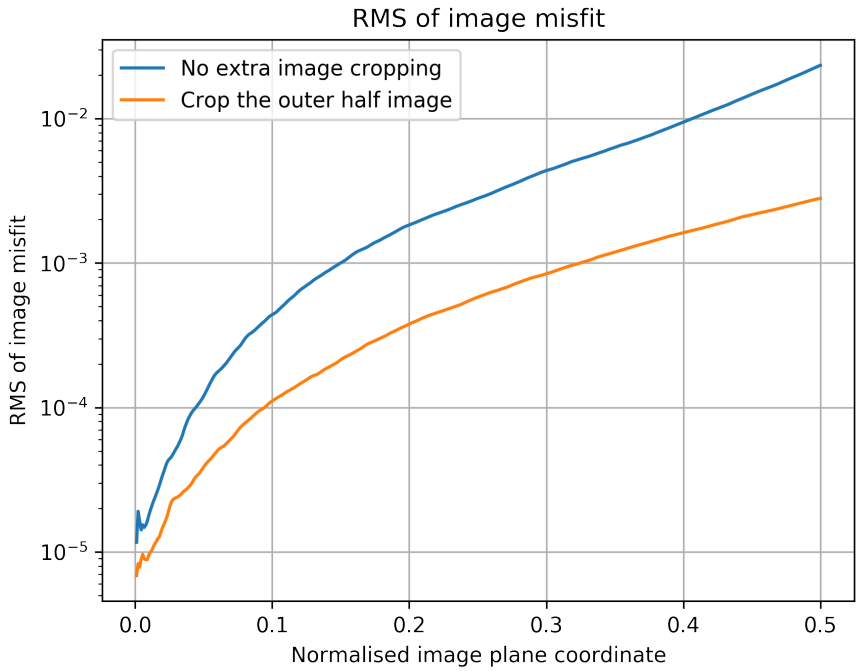


Figure 4.29: RMS value of the image misfit via `WSClean` W–Stacking method is plotted against the normalised image plane coordinate. The case with cropping of the outer half of the double–sized dirty image is plotted as the orange line.

timate the source fluxes at the far corners. Making a larger image with further image cropping can reduce the inaccuracy. This effect is most likely due to more samples on the w–axis when making a larger image. Currently, the `WSClean 2.6` does not provide options to increase the w–plane number manually. Further investigation is needed to work on how many more w–planes are needed to avoid the flux underestimation.

We then add Gaussian noise ($0.1 Jy/Beam$) to the 34-source data, and use `WSClean` to reconstruct the brightness distribution. The reconstructed image is shown in the top map in Figure 4.30. By default, the residuals have been added in. `WSClean` finds the flux of the top right corner source to be 2.72 Jy, which is about 6% smaller. This occurs also for other three corner sources. When we use `WSClean` with a double–sized reconstructed image and then crop the outer half of it, the reconstructed brightness are as shown in the bottom plot. `WSClean` finds the flux of the top right corner source as 2.91 Jy, which is satisfactory.

We conclude that further image cropping can improve tenfold the image misfit when using `WSClean`. It indicates that the number of w–planes suggested by Offringa et al. (2014) is not big enough, leaving the w–axis undersampled. A larger image should be constructed when using current `WSClean` to produce accurate image products, especially at the image edges. Although we cannot directly compare our improved W–Stacking method with the original method in `WSClean` because of the differences in the gridding functions, look-up tables and correcting functions, the accuracy of the celestial sphere dirty image made by `WSClean` is still a concern.

4.5.4 Performance of `CASA` W–Projection method

W–Projection is implemented in `CASA`. Here, we conduct numerical experiments to demonstrate the performance of the `CASA` W–Projection method. We use the same noiseless 34-source data as in Figure 4.1. We choose the W–Projection method by running the `clean` task in `CASA`. The dirty image constructed by `CASA` W–Projection is shown in the left map in Figure 4.31, and the right map is the difference between the dirty image and the DFT dirty image on the celestial sphere. The difference map has a RMS value of 1.49×10^{-2}, far worse than what was obtained using the improved W–Stacking method. There are several reasons for this, including the different choice of gridding function, the use of look–up tables, and the different imaging method used.

We next construct a double–sized dirty image using the `CASA` W–Projection method with cell size unchanged, but retain only the central half of the image. The cropped image is shown in the left map in Figure 4.33. The difference between the cropped dirty image and the DFT dirty image has a RMS value of 7.04×10^{-3}, roughly a two-fold improvement over the case when no extra image cropping is applied.

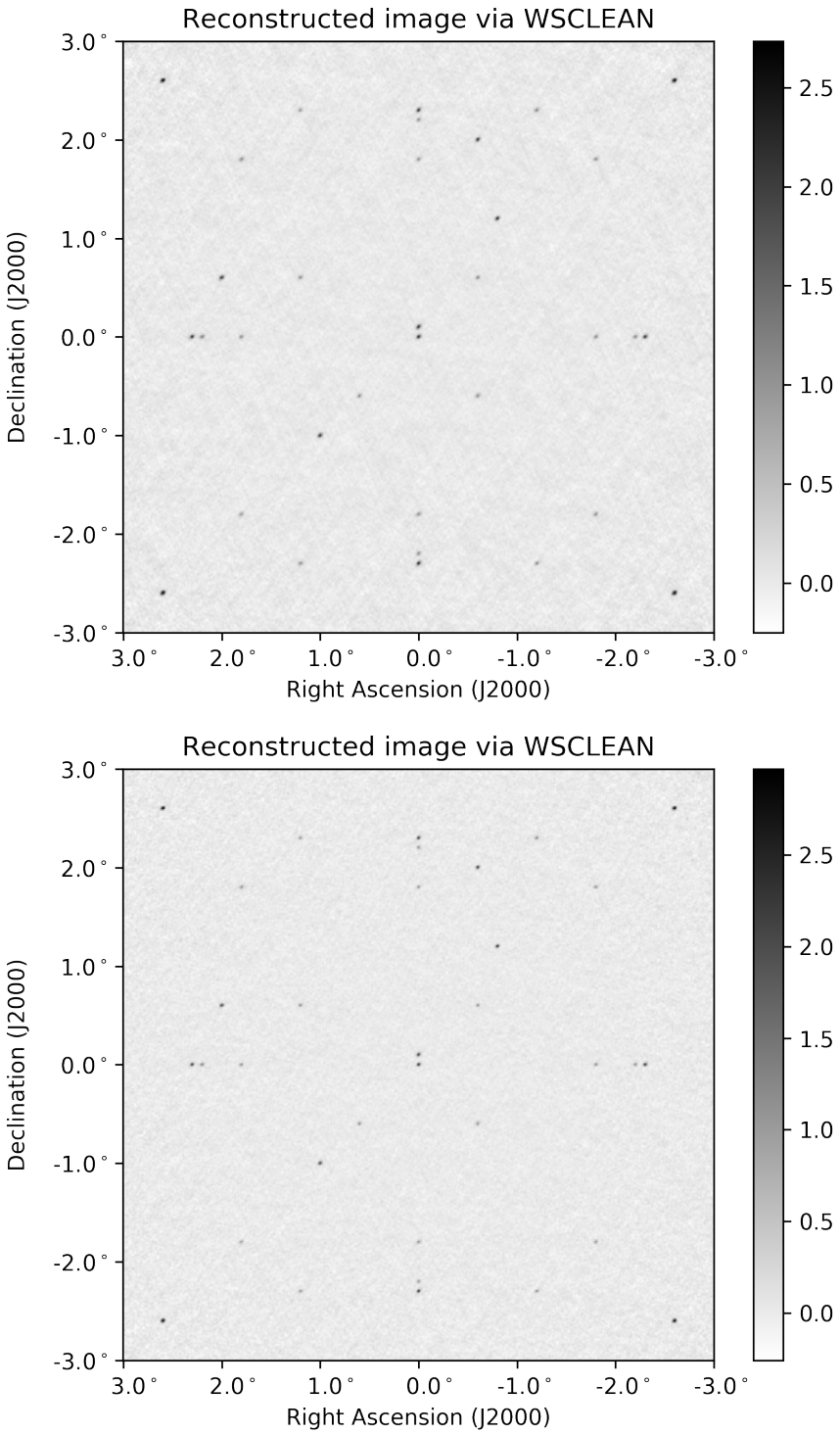


Figure 4.30: Two reconstructed images made by `WSClean`. The upper image is made without any further image cropping, and the lower map is a double–sized reconstructed image with its outer half thrown away. The top right corner source in the upper map has a flux of 2.72 Jy, less accurate than 2.91 Jy found in the bottom map.

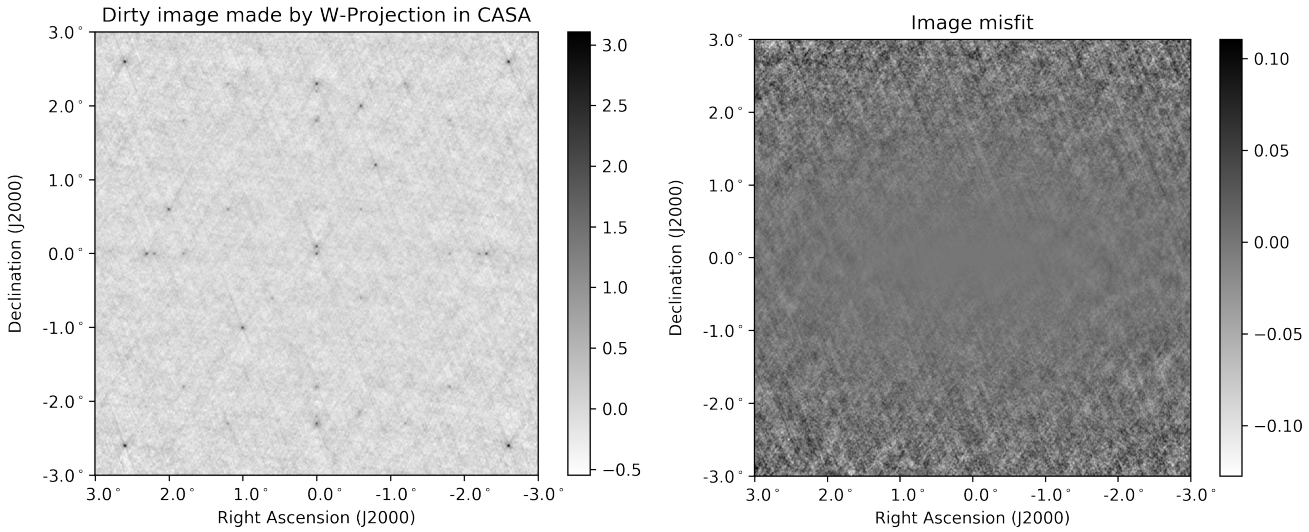


Figure 4.31: Dirty image made through the W–Projection method implemented in `CASA` is shown at left, and its difference from the DFT dirty image on the celestial sphere is shown at right. Primary beam correction has been applied.

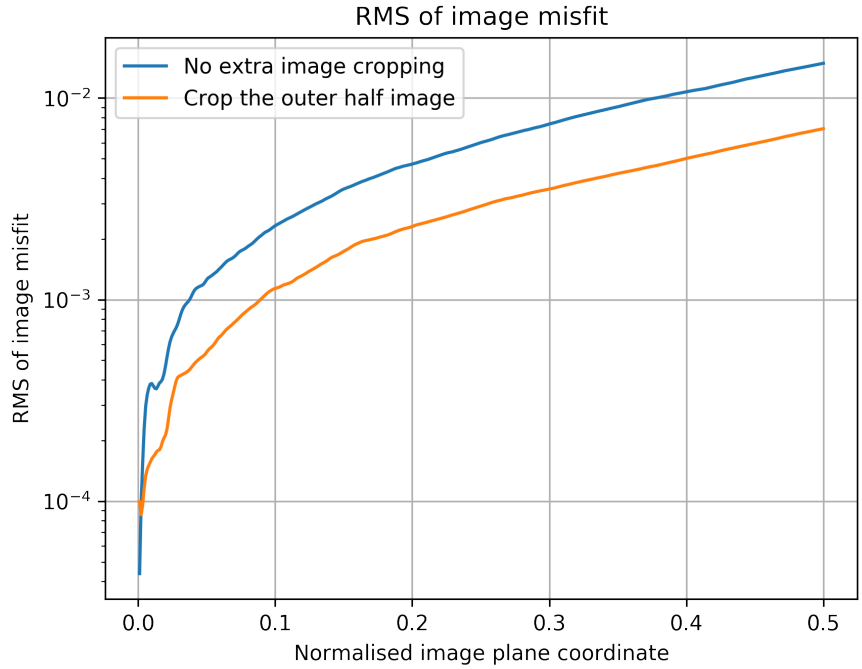


Figure 4.32: RMS value of the image misfit using `CASA` W–Projection is plotted against the normalised image plane coordinate. The case with cropping of the outer half of a double–sized dirty image is shown by the orange line.

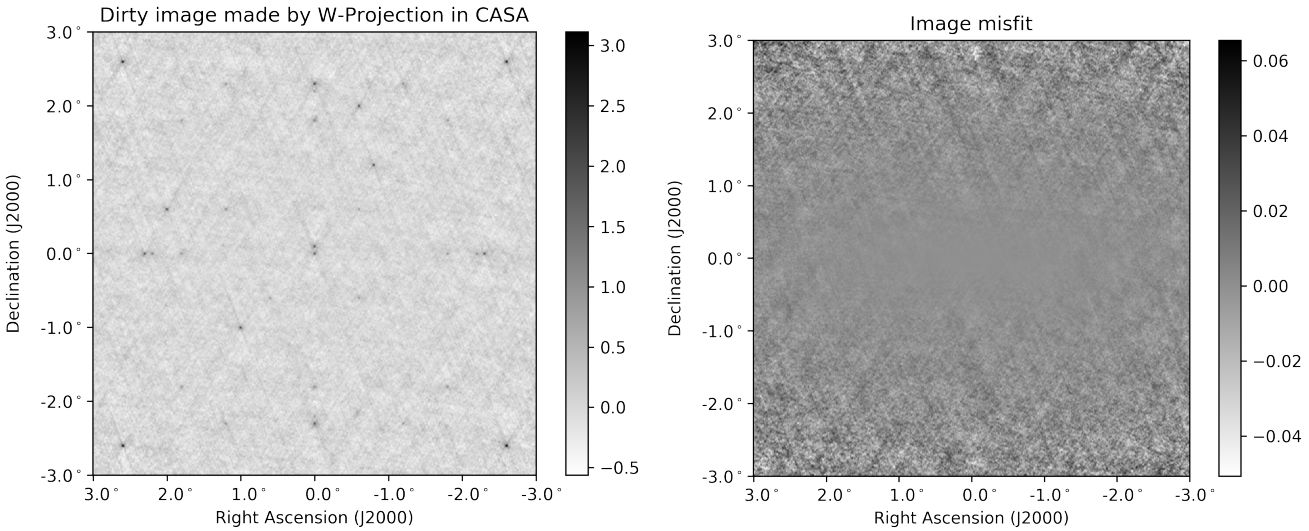


Figure 4.33: Cropped double–sized dirty image made through the W–Projection method implemented in CASA is shown at left, and its difference from the DFT dirty image is shown at right. Primary beam correction has been applied.

We plot the RMS value of the image misfit for both cases against the normalised coordinate in Figure 4.32. In both cases the image misfit levels increase monotonically with the distance from the phase centre. The double–sized dirty image has a smaller image misfit level. We conclude that further image cropping improves the image misfit only slightly. Although we cannot directly compare our improved W–Stacking method with the W-Projection method in CASA because of the differences in gridding functions and look-up tables, the accuracy of the celestial sphere dirty image made by CASA remains a cause for a concern.

We then add Gaussian noise of magnitude $0.1 Jy/Beam$ to the 34-source data, and reapply the W–Projection method in CASA to make a dirty image. The RMS image misfit remains at a magnitude of 10^{-2}. Next, we use the `clean` command in CASA with a threshold of 10 mJy. The Cotton–Schwab CLEAN is chosen. The resulting reconstructed image is shown in Figure 4.34. By default, the residuals have been added in. CASA finds the flux of the top right corner source to be 2.92 Jy, and that of the central sources as 2.07 Jy; these values are adequately accurate.

Although we were previously in doubt about the W–Projection degridding procedure described in Cornwell et al. (2008), see §4.2.3.2, it seems that CASA works well on our simulated data. We need to further investigate the implementation of W–Projection method.

4.6 Computational cost

In this section we compare the computational cost of the original and improved W–Stacking methods. As calculated in Section 4.3, the overall computational cost for the improved W–

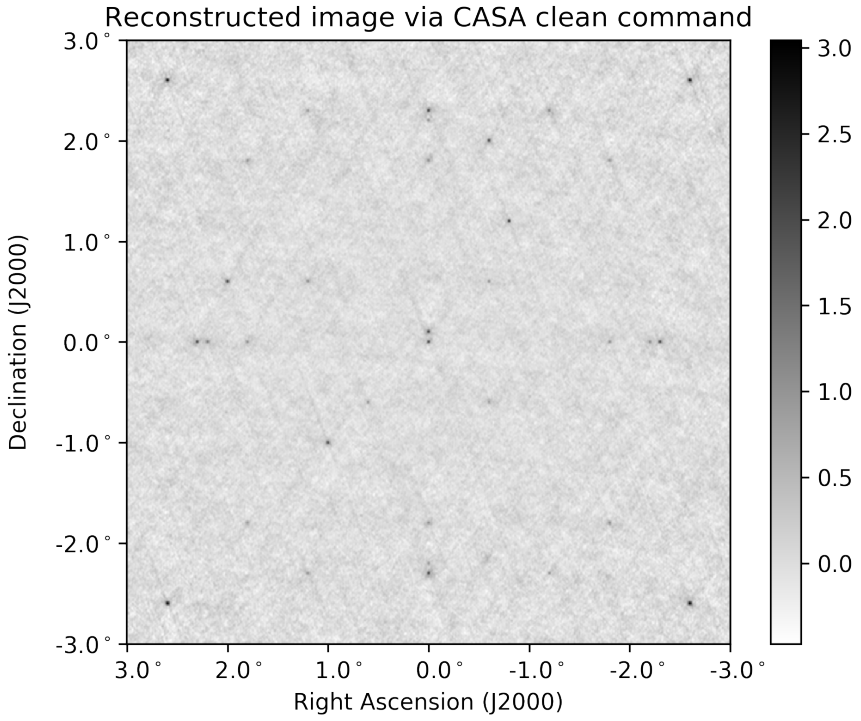


Figure 4.34: Image reconstructed via Cotton–Schwab CLEAN from a dirty image made using the W–Projection method implemented in CASA. Primary beam correction has been applied.

Stacking method is listed in Table 4.3. The original W–Stacking method has similar operational cost apart from a difference in the gridding operations, where only two–dimensional gridding is undertaken. We assume that the original W–Stacking method also requires the correcting function in all three dimensions, although we do not claim to know how the correcting function $h(n)$ should be chosen. We denote the number of w–planes for the original and improved W–Stacking methods by N_w and $N_{w'}$ respectively.

Table 4.3: Overall computational cost of the original and improved W–Stacking methods

Method	Computational cost
Original W–Stacking	$2N_v W^2 + \left[\frac{N_w}{4x_0^2}\log\left(\frac{N_x N_y}{4x_0^2}\right) + 2N_w + 2\right] N_x N_y$
Improved W–Stacking	$2N_v W^3 + \left[\frac{N_{w'}}{4x_0^2}\log\left(\frac{N_x N_y}{4x_0^2}\right) + 2N_{w'} + 2\right] N_x N_y$

Notice that N_w and $N_{w'}$ are different because of differing calculation formulas.

We first study the relation between the computational cost and the number of w–planes. For a typical wide–field observation, we set the number of visibilities N_v to be 10^7. The least–misfit gridding function with $W = 7$ and $x_0 = 0.25$ is used to achieve the single precision limit in the image misfit level. The size of the image is set at $N_x = N_y = 2560$. Even though N_w and $N_{w'}$

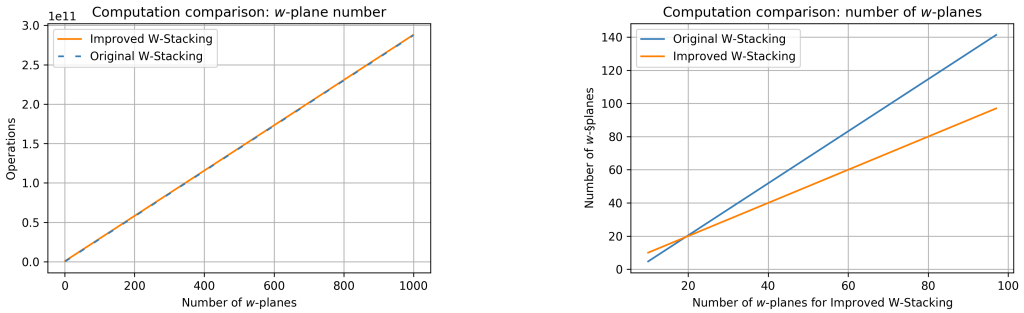


Figure 4.35: Left: computational cost comparison with differing numbers of w–planes, with all other parameters the same for the original and improved W–Stacking methods. Right: comparison between the number of w–planes involved in the two different methods.

have different values for the same dataset, we set them equal and change their values from 1 to 1000. Figure 4.35 (left graph) plots the number of operations needed for the two methods versus the number of w–planes.

This plot shows that the computational cost increases linearly with the number of w–planes. When the same number of w–planes is used in each method, the original W–Stacking has 5.88×10^8 fewer operations than the improved method because there are fewer gridding operations. The difference is not easy to distinguish by eye, as the total number of operations sums to a value as large as 10^{11}.

In practice, N_w and $N_{w'}$ are chosen with different values, and we must therefore compare N_w and $N_{w'}$ using the same dataset. If we take $N_w = 2\pi(w_{\max} - w_{\min}) \max_{l,m}(1 - \sqrt{1 - l^2 - m^2})$ following Equation 4.26, and $N_{w'} = \frac{\max_{l,m}(1-\sqrt{1-l^2-m^2})(w_{\max}-w_{\min})}{x_0} + W$ following Equation 4.30 then, since we assume the two methods are applied with the same dataset, we have

$$N_{w'} = \frac{N_w}{2\pi x_0} + W$$

where $W = 7$. This relation is plotted at right in Figure 4.35; N_w is plotted against $N_{w'}$ in blue line, while $N_{w'}$ is of course plotted against itself. We see that $N_{w'} < N_w$ when $N_{w'} > 19$, which implies that the improved W–Stacking method uses fewer w–planes than the original method.

For $N_{w'} \leq 19$, the corresponding value of N_w is not greater than $N_{w'}$. Because there are fewer gridding operations, the computational cost for the original W–Stacking method is less than for the improved method. An example can be seen in the top left graph in Figure 4.36, with $N_w = 15$ and $N_{w'} = 17$. The computational cost is plotted against the number of visibilities, ranging from 10^5 to 10^7. We conclude, unsurprisingly, that the larger the dataset is, the higher the computational cost involved.

We have to bear in mind that `WSClean` needs more than N_w w–planes to avoid the flux underestimation at the far corners. Therefore, it is very likely that N_w is always more than $N_{w'}$

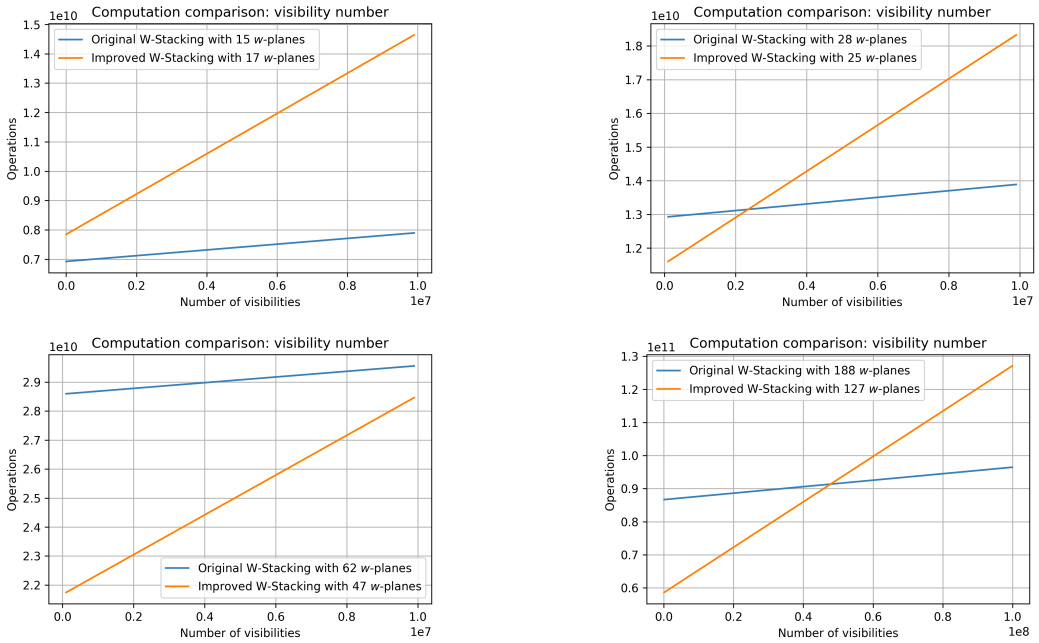


Figure 4.36: Computational comparisons between the original and improved W–Stacking methods with differing values of N_w and $N_{w'}$ against the number of visibilities. The image size is fixed to 2560 by 2560.

so as to achieve the full potential of the original W–Stacking method.

When $N_{w'} > 19$, for example $N_{w'} = 25$ and $N_w = 28$ (as in the top right graph in Figure 4.36), the computational cost of the original W–Stacking method is larger than the improved method when the visibility number is less than 2.3×10^6; otherwise, the improved method involves more operations.

For a dataset that requires at least $N_{w'} = 47$ and $N_w = 62$, the original method incurs greater computational cost for the given data size range; see the bottom left graph in Figure 4.36

For a more extreme wide–field case in which $N_{w'} = 128$ and $N_w = 189$ are needed, if the dataset has a visibility number less than 4.8×10^7, the improved W–Stacking method is cheaper computationally.

We can conclude this set of comparisons as follows:

- when the number of w–planes needed in the improved W–Stacking method is greater than in the original method, the improved W–Stacking method has higher computational cost.
- the extra gridding expenses in the improved W–Stacking method can be compensated by the decreased w–plane number. When the image size is fixed, the extent of compensation depends on the w–plane number and visibility number, and the w–plane number is in turn determined by the range of w and the size of the field.

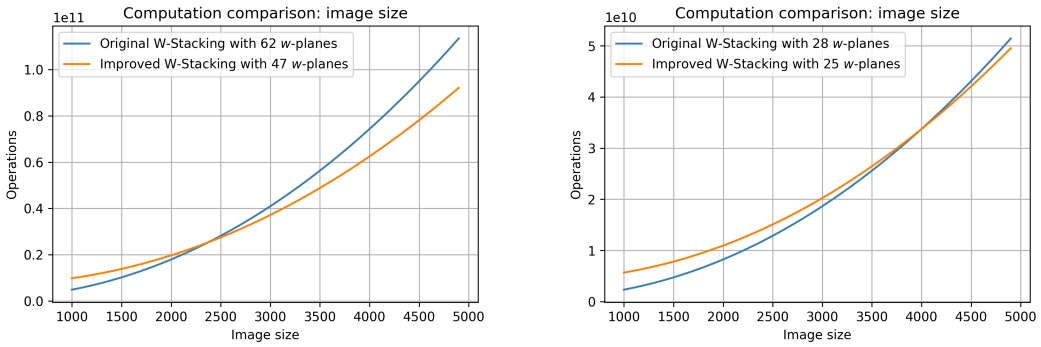


Figure 4.37: Computational comparisons between the original and improved W–Stacking methods with differing values of N_w and $N_{w'}$ against the image size. The number of visibilities is fixed to 10^7 and 6×10^6 respectively.

We then look at the relation between the computational cost and the image size. The image size is changed from 1000 to 5000 for a wide–field dataset that requires $N_{w'} = 47$ and $N_w = 62$. The computational cost comparison between the two methods are demonstrated in the left plot in Figure 4.37 with the visibility number fixed to 10^7. The improved W–Stacking method becomes computationally cheaper than the original method when the image size is larger than 2400.

We repeat the same experiment but change $N_{w'}$ and N_w to 25 and 28 respectively. In the top right graph in Figure 4.36, with a visibility number of 6×10^6 and an image size of 2560 by 2560, the original W–Stacking method is computationally cheaper than the improved W–Stacking method. When the image size increases to more than 4000, as shown in the right plot in Figure 4.37, the improved W–Stacking method becomes computationally cheaper instead. The visibility number is fixed to 6×10^6.

We conclude this set of comparisons as follows:

- the extra gridding expenses in the improved W–Stacking method can be compensated by the decreased w–plane number. When the visibility number is fixed, the extent of compensation depends on the w–plane number and image size.
- with the increase of the image size, the computational cost of both methods becomes primarily dominated by the increased FFT operations. With a smaller w–plane number required, the improved W–Stacking method can be computationally cheaper than the original method when a larger image is to be made.

It is impossible to state that one method is always more computationally expensive. The computational cost is dependent on the dataset, especially its visibility number, the range of w and the size of the field, as well as the image size. For our 34-source simulated data, the original and improved W–Stacking methods respectively require at least 47 and 37 planes.

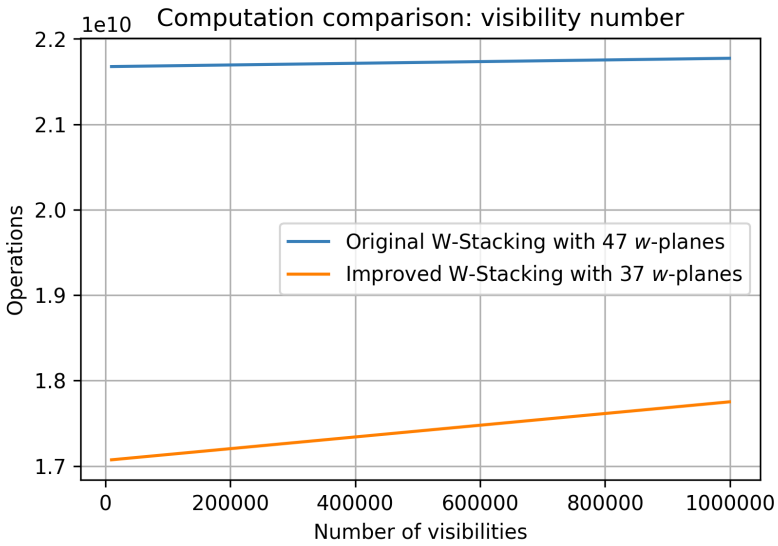


Figure 4.38: For our 34-source simulated data with 46332 visibilities, it proves computationally cheaper to use our improved W–Stacking method. The original and improved W–Stacking methods respectively require at least 47 and 37 planes.

Since the simulated data has 46332 visibilities, it is cheaper computationally to use our improved W–Stacking method with an image size of 2560 by 2560[i]; see Figure 4.38.

4.7 Conclusions

In this chapter, we began by explaining that the significant w–term is the source of wide–field imaging problems. The w–term cannot be neglected when the field of view is not narrow, nor when values of w are large. In these cases, sources far from the phase centre become blurred on the image plane when a two–dimensional Fourier transform is performed. The Fresnel number was introduced in order to judge whether the w–term requires careful consideration for a particular observation.

The wide–field imaging problem arises not only when non–coplanar arrays are in use, but also when planar arrays are used with large values of w. Experiments have shown that for a planar array whose values of w and v are almost perfectly correlated, in the linear relation $w = \alpha v$, the dirty image of the celestial sphere simply undergoes a displacement from the dirty image of the tangent plane by $-\alpha n$ along the l–axis.

We then introduced several existing wide–field imaging methods, including the three–dimensional Fourier transform method, the polyhedron imaging method, the W–Projection method and the W–Stacking method. Use of the tapering function in the W–Projection method

[i]This remains true even when the image size decreases to 900.

is problematic. The introduction of the tapering function is confusing and the imaging steps given by Cornwell et al. (2008) are questionable. We provide the imaging procedures for W–Projection with use of the gridding function and correcting function, which are consistent with that suggested in Venkata (2010). The least-misfit gridding function is recommended for better accuracy.

We proposed a revised wide–field imaging method based on the three–dimensional Fourier transform method, named 'N–Faceting'. This involves the imaging of multiple n–planes followed by a three–dimensional deconvolution process, where multiple position–independent beams are in use. In order to obtain the final reconstructed image, the reconstructed three–dimensional image cube is projected onto the tangent plane, followed by an convolution with a two–dimensional restoring beam. We applied this method to our 34–source simulated data, and displayed the dirty images and residuals by facet as well as the reconstructed images.

An improved W–Stacking method has been proposed, based on the original W–Stacking method. Instead of two–dimensional gridding on w–planes, a three–dimensional gridding is applied. This improvement makes it possible to calculate and apply the correcting function in the n–dimension. The least–misfit gridding and correcting function pair with $W = 7$ and $x_0 = 0.25$ is recommended in order to achieve the single precision limit on the misfit between the DFT and FFT dirty images on the celestial sphere. In our numerical experiment with the 34-source simulated wide–field data, the image misfit reaches the double precision limit when using least–misfit gridding and correcting functions with $W = 14$ and $x_0 = 0.25$.

We also proposed a new way to calculate the number of w–planes needed, which is determined by the field size and the w range of the given dataset. In our application, the number of w–planes calculated is less than that required by the original W–Stacking method, resulting in fewer FFT operations. The use of three–dimensional gridding in the improved W–Stacking method increases the corresponding gridding computational cost, however. Following a comparison of the computational cost of the original and improved W–Stacking methods, we conclude that the computational cost is dependent on the dataset, especially its visibility number, the range of the field of view and the w range, as well as the image size. The size of the image impacts the FFT cost. For our 34-source simulated data, the improved W–Stacking method is computationally cheaper.

In comparison, the image misfit levels achieved by the W–Projection algorithm in `CASA` and the original W–Stacking algorithm in `WSClean` are not less than 10^{-3} in magnitude. The dirty images and reconstructed images constructed via `WSClean` underestimate the fluxes of sources located far from the phase centre. This limitation is overcome by increasing the image size, which indirectly increases the number of w–planes, indicating that the w–axis is undersampled. Therefore, more w–planes than the default recommendation should be used in order to avoid

the flux underestimation, resulting in even more FFT computational cost than the improved W–Stacking method.

We intend imminently to develop and apply the N-Faceting method. Also, how many more w–planes than the default recommendation should be used for `WSClean` needs further investigation. If possible, we wish to propose that `WSClean` should take up our improved W–Stacking method as an option. If `BaSC`, introduced in Chapter 2, were generalised to wide–field applications, the accuracy of the dirty images on the celestial sphere would be vitally important. We hope to test whether the accuracy improvement on the dirty image would increase the source detection accuracy.

Chapter 5

Two real data tests in wide-field imaging

This chapter applies the improved W–Stacking method to two sets of real data. The first dataset comprises a VLA D–array observation of the supernova remnant G55.7+3.4 made on August 23, 2010. The data were downloaded and calibrated according to the instructions at the CASA website https://casaguides.nrao.edu/index.php/EVLA_Wide-Band_Wide-Field_Imaging:_G55.7_3.4-CASA4.4.

The second dataset comprises GMRT data observed at frequencies ranging from 32 MHz to 610 MHz in 24 channels. This data was used by Whittam et al. (2017), and its wide-field imaging was performed in AIPS using multiple image facets. Details of the data reduction and related research are given in Whittam et al. (2017). Below, we use this calibrated data to test our improved W–Stacking method.

5.1 Results of the Improved W–Stacking method on G55.7+3.4 data

The L–band[a] observation is targeted at the supernova remnant G55.7+3.4 centred on ($19^h21^m40^s$, $+21°45'$). The VLA data for the CASA tutorial has already been preprocessed from the 170 GB raw observations down to about 45 GB in size by means of operations such as online flagging and time averaging. Further calibration is still needed, however, for which we followed the instructions given in the tutorial. Of 7343848 visibility entries, only 21.6% were retained (1583435 visibilities) with four spectral windows out of eight. Each spectral window has 64 channels.

[a]The L–band covers the frequency range 1-2 GHz.

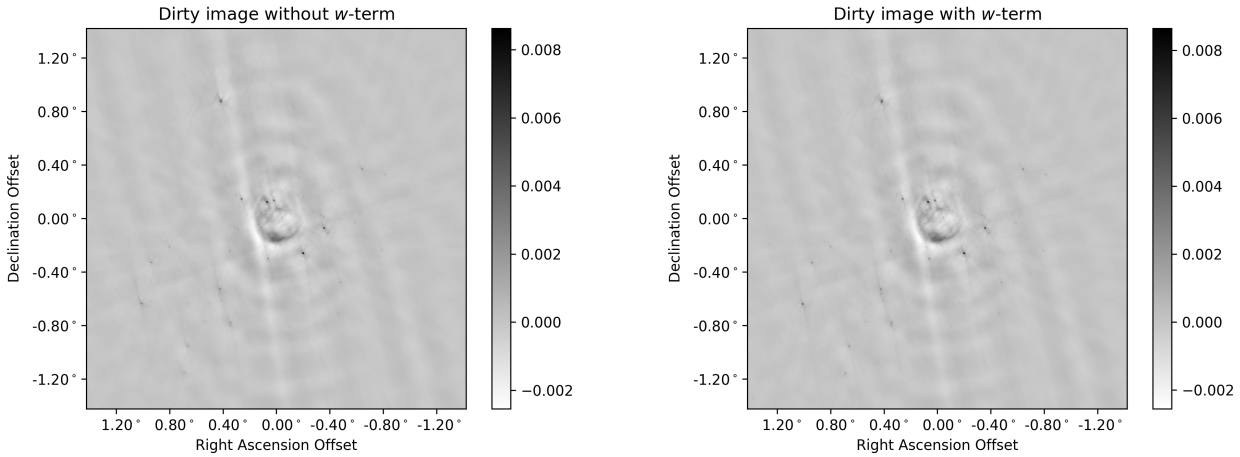


Figure 5.1: Left: tangent plane dirty image of the VLA data using CASA. Right: celestial sphere dirty image of the VLA data using CASA. The field of view is centred on ($19^h21^m40^s$, +21°45′). The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

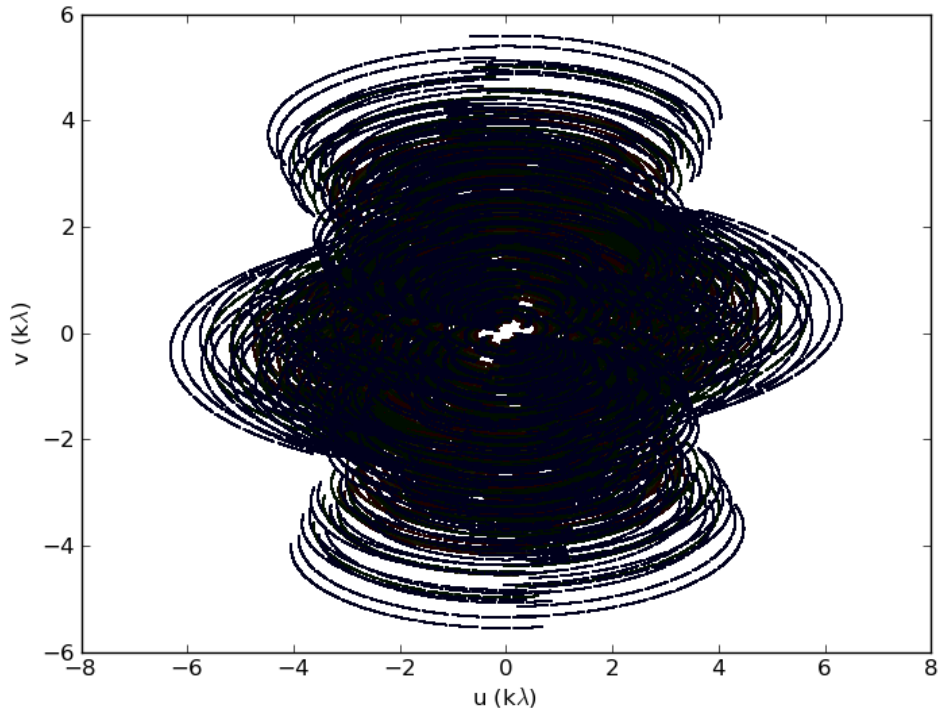


Figure 5.2: uv–coverage of the calibrated data. Both axes are in units of kilowavelengths. There are large gaps in the coverage at the centre, giving rise to a negative ring around the supernova remnant and the unexpected stripes that are visible.

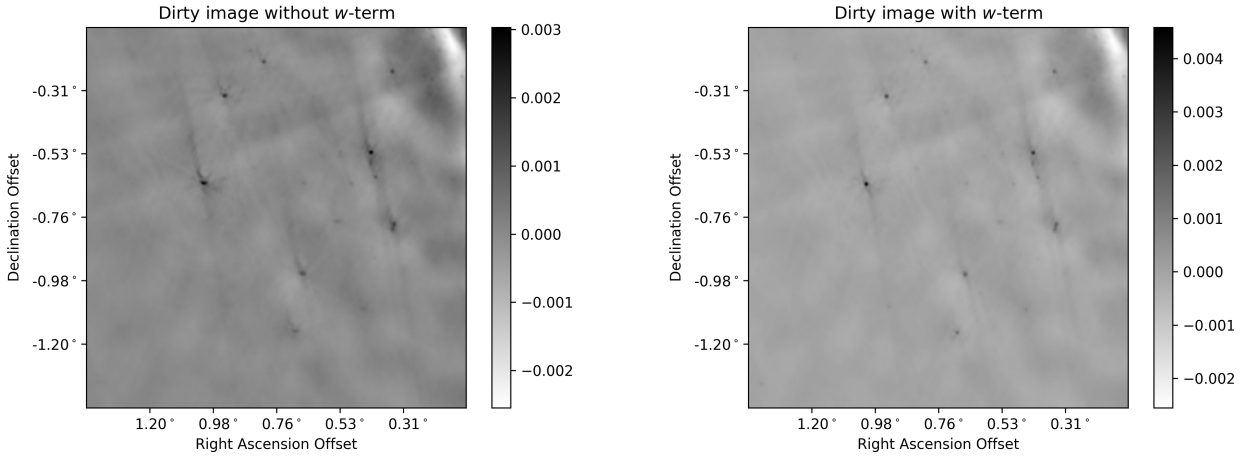


Figure 5.3: Point sources far from the phase centre are visible with non-coplanar effects (such as prominent arcs) in the enlarged tangent plane dirty image at left. Use of W–Projection removes the artefacts, as shown at right.

We first use the W–Projection method implemented in `CASA` to construct a tangent plane dirty image and a celestial sphere dirty image, as shown in Figure 5.1. Natural weighting is used. The image size is 2560 by 2560, with a pixel size of 4 arcsec. The field of view is centred on $(19^h21^m40^s, +21°45')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

There is a large gap in the uv–coverage at the centre, as shown in Figure 5.2. This gives rise to negative ring–like artefacts around the supernova remnant and the stripes visible in the figure. Since Briggs weighting makes the uv-coverage more uniform (Briggs 1995), it can smooth out the stripes, as demonstrated within the tutorial (although this is not the focus of the present thesis).

If we zoom in the bottom left portion of both images, shown in Figure 5.3, then we see in the tangent plane dirty image (at left) non-coplanar effects of the processing, in particular prominent arcs around the point sources far from the phase centre. Use of W–Projection method removes the artefacts, as seen on the right map.

We next apply the improved W–Stacking method to the same calibrated data. All four spectral window data has been imaged. For each image, the fluxes have been adjusted with a spectral index of 0.5. The resulting averaged celestial sphere dirty image is shown in Figure 5.4, where natural weighting has been used. The image size is 2560 by 2560, with a pixel size of 4 arcsec. The field of view is also centred on $(19^h21^m40^s, +21°45')$. The image is shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

The map at right is an enlargement of the lower left corner of the left map. The improved

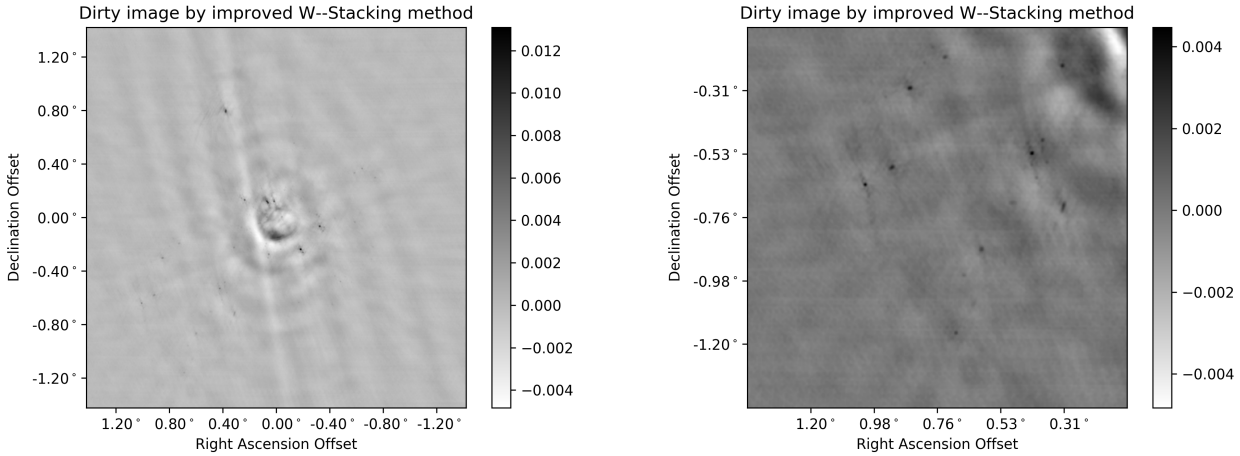


Figure 5.4: Left: celestial sphere dirty image made using the improved W–Stacking method. Right: Zoomed-in window of the bottom left portion of the dirty image. The field of view is centred on $(19^h21^m40^s, +21°45')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

W–Stacking method successfully removes the non-coplanar effects.

To cover the full observation frequency range, the improved W–Stacking method requires 19 w–planes , whereas the original W–Stacking method embedded in `WSClean` requires 24 w–planes when handling the same data. A calculation of the computational cost of the original and improved W–Stacking methods, as in Section 4.6, reveals that the original W–Stacking method is computationally cheaper by 5.73×10^{10} operations, since the increased gridding cost is greater than the reduced FFT cost. The high visibility number (64 channels of 1583435 visibilities) contributes to this far greater gridding cost.

Since the improved W–Stacking method generates more accurate celestial sphere FFT dirty images (as explained in Chapter 4), it is still recommended in order to achieve the single precision limit in the image misfit. For observations with larger ranges of w and wider fields, the improved W–Stacking method may be less computationally intensive because of the further reduction in w–planes.

5.2 Results of the Improved W-Stacking method on GMRT data

We obtained the 610-MHz calibrated GMRT data from Dr. Dave Green. It contains 415613 visibilities in a single spectral window with 24 channels. The observations cover the field AMI001 of 0.84 deg^2 centred on $(00^h23^m10^s, +31°53')$. The size of the dirty beam is 7.6×7.3 arcsec (Whittam et al. 2017). Rather than use image facets, we apply the improved W–Stacking method to the data. The tangent plane dirty image and the celestial sphere dirty image are

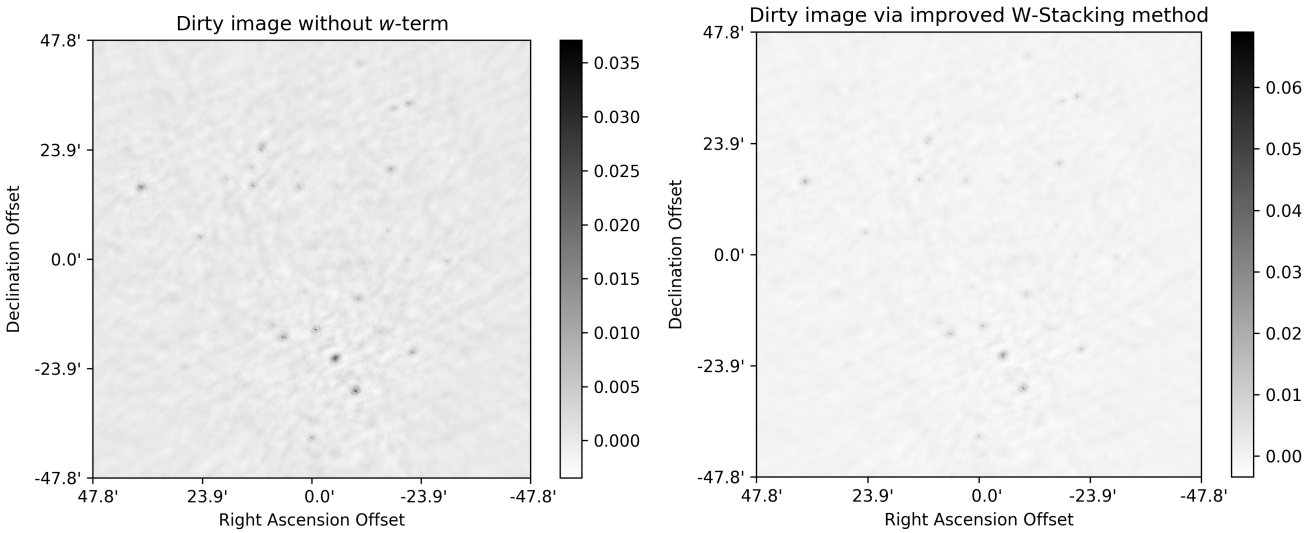


Figure 5.5: Left: tangent dirty image of GMRT data. Right: celestial sphere dirty image made by the improved W-Stacking method. The field of view is centred on $(00^h23^m10^s, +31°53')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

shown in Figure 5.5 at left and right. Both images are 4096 by 4096 in size, with a cell size of 1.4 arcsec. The field of view is centred on $(00^h23^m10^s, +31°53')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

Only a few radio objects can be seen by eye in these dirty images, because the field of observation contains many faint radio sources whose fluxes are too small to stand out from the bright ones. Later, we shall exhibit some areas containing faint radio sources.

In Figure 5.6 we zoom in on the top left portion of both images. In the tangent plane dirty image displayed at left, we see non-coplanar effects such as prominent arcs around the two nearby point sources which are far from the phase centre. These artefacts make it impossible to confirm the existence of the second point source. Use of W–Stacking method removes the artefacts, permitting the two point sources to be distinguished (as in the right map).

The method required 40 w–planes because of more significant w values, whereas `WSClean` requires 51 w–planes to deal with the same data.

A calculation of the computational cost of the original and improved W–Stacking methods, as in Section 4.6, reveals that the original W–Stacking method is computationally more expensive (by 7.81×10^9 more operations) with this image size. This is because the reduced FFT cost more than compensates for the increased gridding cost. The improved W–Stacking method is therefore preferable in both computational cost and accuracy.

We made the dirty image using the improved W-Stacking method on a computer with a 6-core Intel i7-4930 processor and 12GB memory. We used the least-misfit gridding function

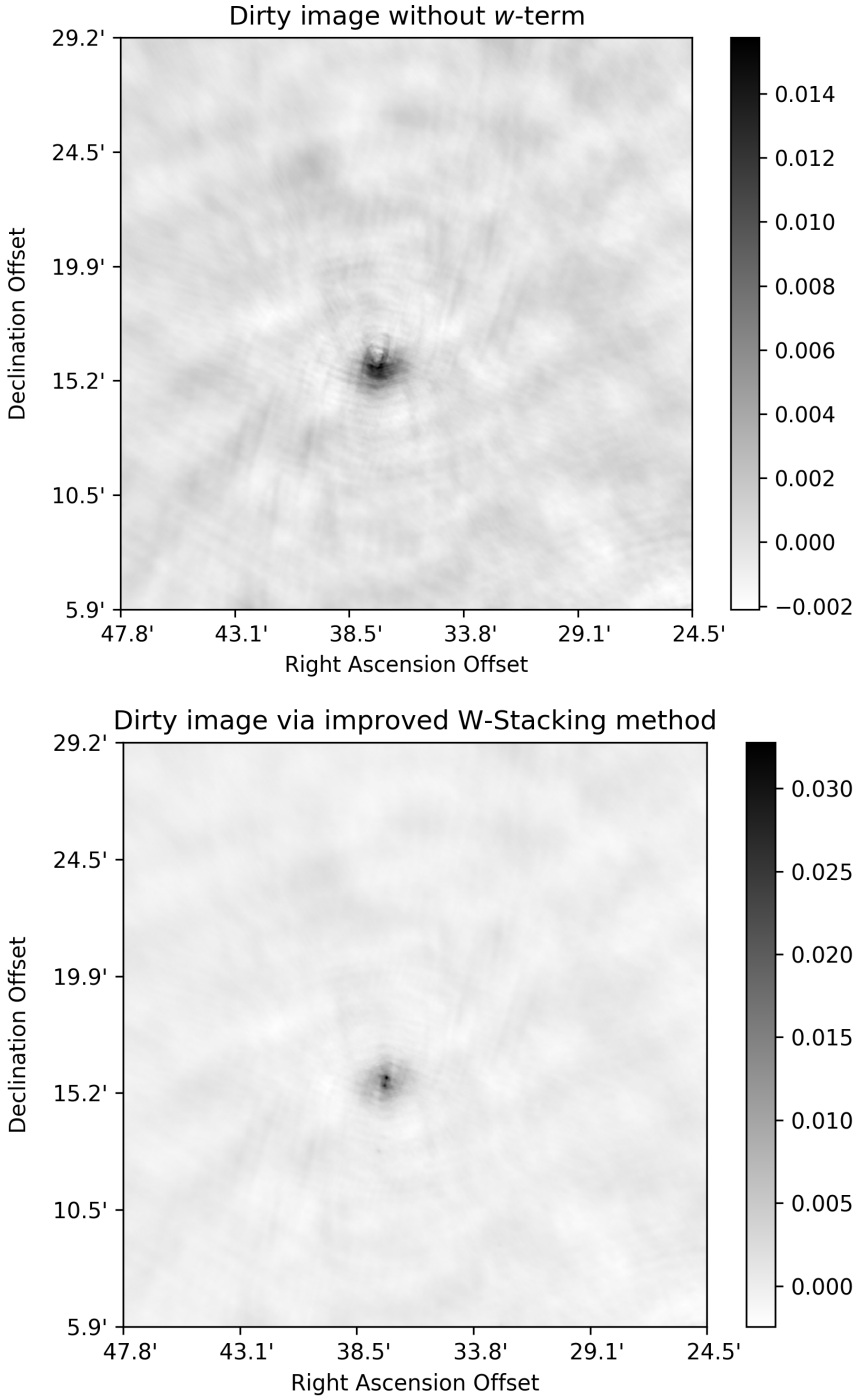


Figure 5.6: Two nearby sources far from the phase centre are shown with non-coplanar effects (such as prominent arcs) in the enlarged tangent plane dirty image at the top. The W–Stacking method removes the non-coplanar effects, as shown on the lower map, and the two sources are resolved.

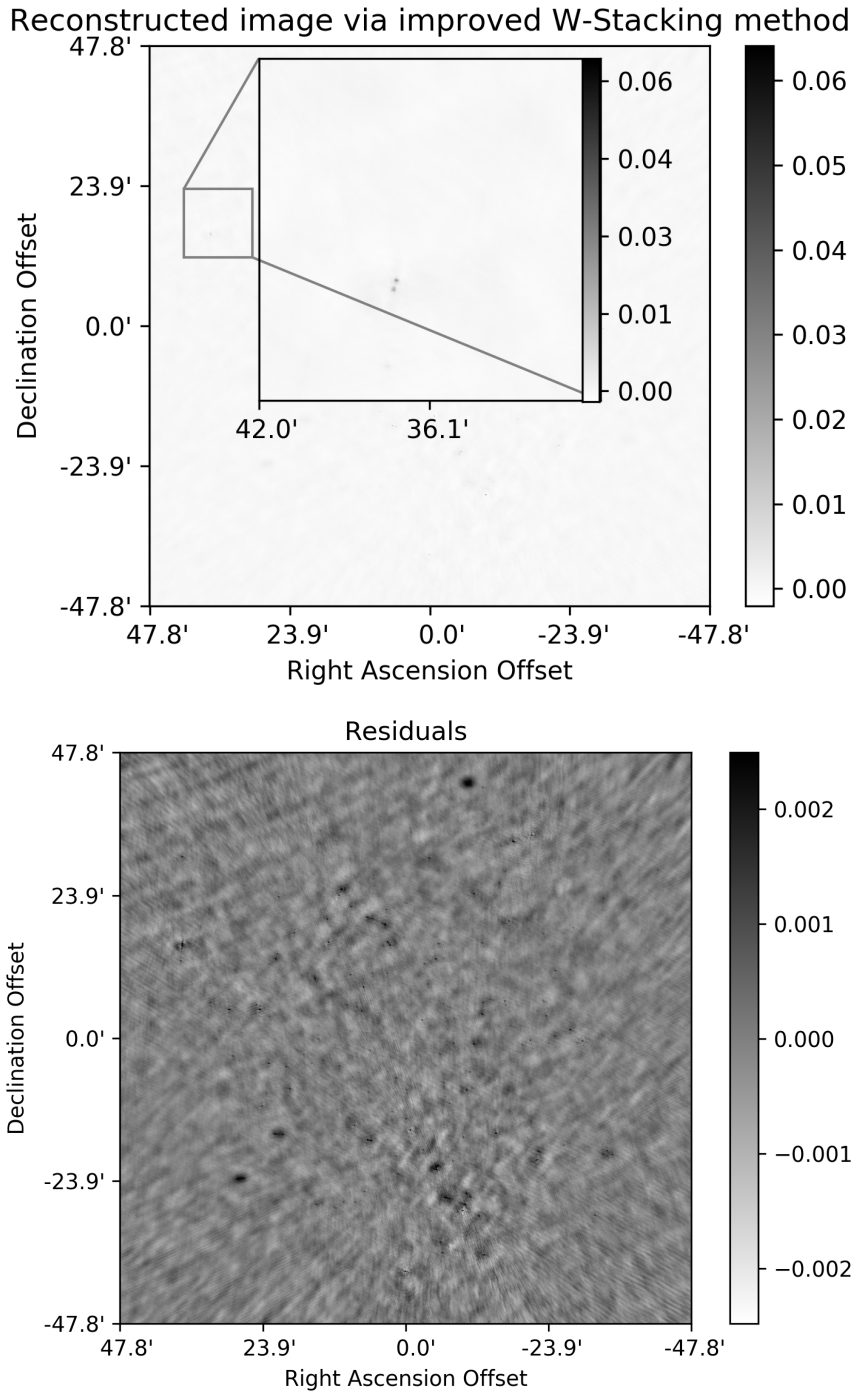


Figure 5.7: Sky brightness reconstructed via Högbom CLEAN (top), and the corresponding residuals (bottom). Many calibration errors occur in the residual image. The field of view is centred on $(00^h23^m10^s, +31°53')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

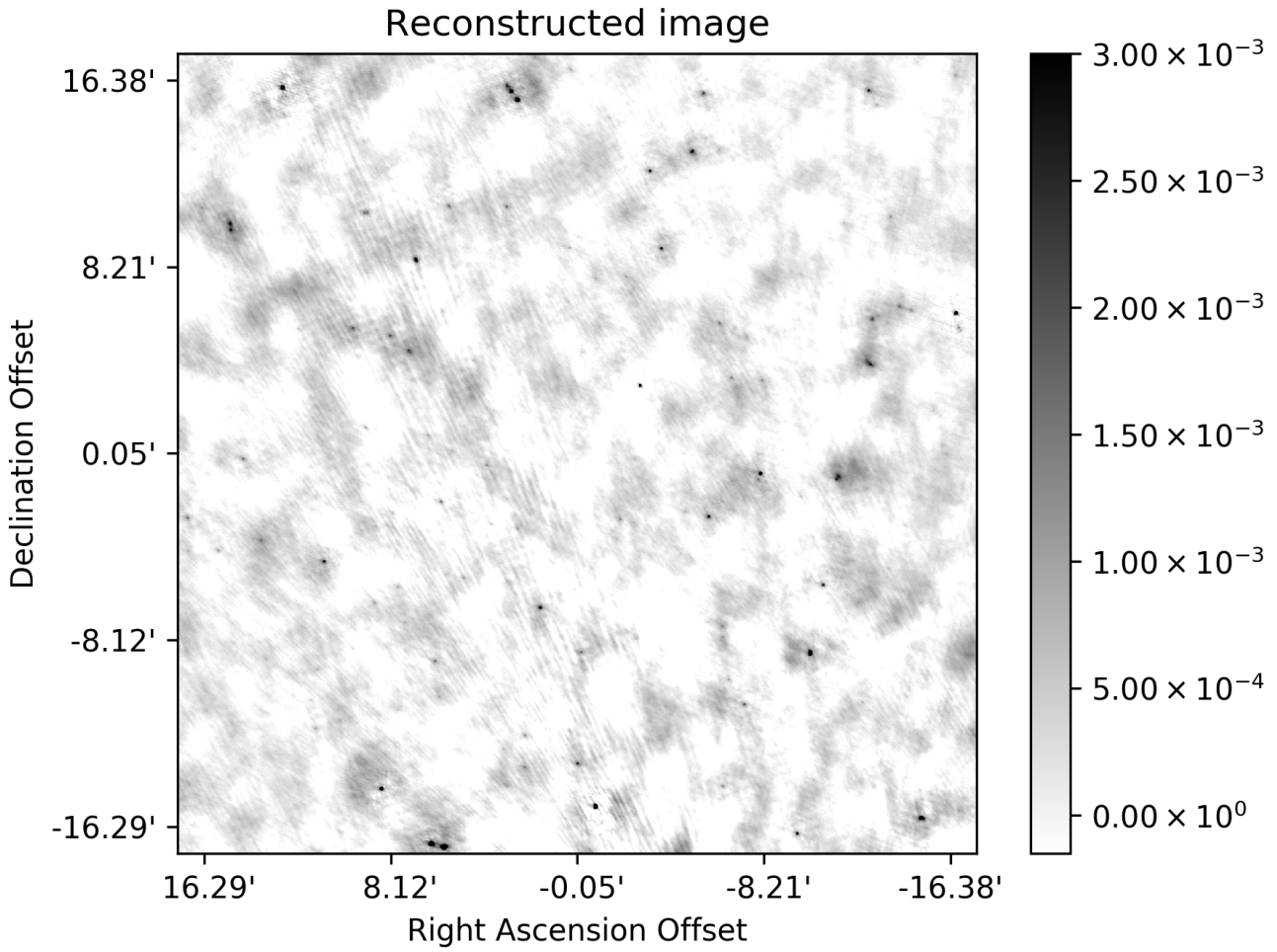


Figure 5.8: The central half of the reconstructed image. No primary beam correction has been applied.

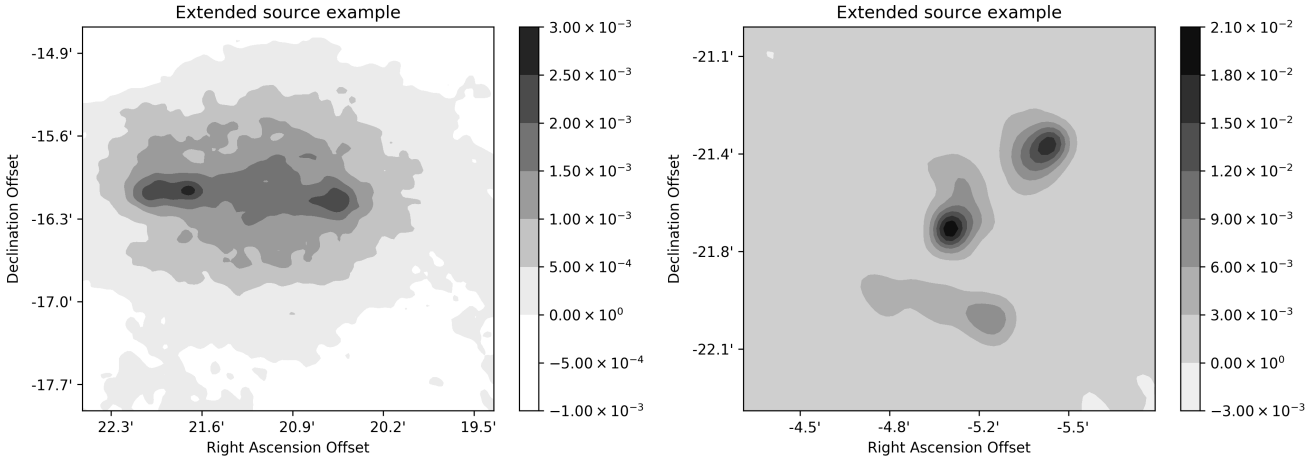


Figure 5.9: Two examples of extended sources in the image reconstructed using the improved W–Stacking method.

($W = 7$ and $x_0 = 0.25$) without a look-up table, and 10 seconds were taken to calculate the weights for all visibility data. The gridding process then took 19 seconds. The FFT operation took the longest clock time, of 304 seconds. This is because 40 w-planes involves 40 FFT operations. By taking advantage of the multiple cores, the FFT time was successfully reduced to 114 seconds.

A simple Högbom CLEAN was implemented to reconstruct the sky brightness from the dirty image made using the improved W–Stacking method. In (Whittam et al. 2017), 955 sources were detected, but most of these are very faint, and there are a few sources that are 10 or even 100 times brighter in the observational field. Because of this great contrast, most sources can scarcely be seen. The reconstructed image is shown in the top map in Figure 5.7, and its corresponding residuals are below. Many calibration errors occur in the residual image. The field of view is centred on $(00^h23^m10^s, +31°53')$. The two images are shown with the right ascension offset and declination offset as x-axis and y-axis respectively.

To demonstrate more clearly the two bright sources shown in Figure 5.6, we zoom into the area by a factor of 5. The two distinct point sources are clearly visible.

In Figure 5.8 we display the central half of the field, with a different contrast. No primary beam correction has been applied. Most sources visible in the reconstructed image of the same area[b] in (Whittam et al. 2017) can be seen in Figure 5.8, except for some of the faintest ones. This is probably because we have not CLEANed the dirty image in sufficient depth. This limitation will be dealt with in future papers that are under draft at the present time of writing.

Figure 5.9 shows two examples of extended sources detected in the reconstructed image. These have the same shape as in (Whittam et al. 2017).

5.3 Conclusions

The improved W–Stacking method has been applied to two real wide–field observations. The celestial sphere dirty images and the reconstructed images were determined and are shown. Comparison between the tangent plane and celestial sphere dirty images demonstrates that the improved W–Stacking method successfully removes non-coplanar effects that occur due to the w–terms.

We see also that for some wide–field observations, such as the VLA observation of G55.7+3.4, the w–planes required by `WSClean` are not very much more than in the improved W–Stacking method. The computational cost of the improved W–Stacking method is then larger, because the increased cost of gridding is greater than the reduced cost in FFT. Since the improved W–Stacking method can produce more accurate celestial sphere FFT dirty images,

[b] the two areas are not strictly identical, but are very close to each other.

as explained in Chapter 4, we still recommend the improved W–Stacking method if the single precision limit in the image misfit is demanded.

In contrast, for observations such as the GMRT observation on the field AMI001, the w–planes required by `WSClean` greatly exceed those needed by the improved W–Stacking method. The reduced FFT cost then more than compensates for the increased gridding cost, and the improved W–Stacking method is superior in both computational cost and accuracy.

In summary, these two experiments confirm that the improved W–Stacking method works well on real observational data. For observations with a larger w range, the improved W–Stacking method is recommended in order to obtain a more accurate image and lower computational cost.

Chapter 6

Conclusions and Future work

The stated aim of this thesis was to develop accurate imaging approaches during the imaging process in radio interferometry. We focused on three different applications, namely source extraction, the gridding and degridding process, and wide–field imaging. The aim in every case was to improve the accuracy of the outputs of these processes, whether dirty images or source extraction results. The first two research projects commenced as a furthering of the work in Tan (1986)'s PhD. To this end we used the Bayesian multiple-source model and the gridding criteria proposed by Tan (1986).

In Chapter 2 we intended to propose a method which can work directly on dirty images, since CLEANed images are not a true representation of sky brightness and the reconstruction process takes much time and human intervention. We also wished to address the problem that the resolving power is limited by the size of the CLEAN restoring beam, yet science of interest occurs at such scales. We started from the Bayesian multiple-source model proposed in Tan (1986), then implemented the model utilising a MCMC driver `BayeSys`, followed by a choice of appropriate clustering algorithms. This approach has been embedded in a `Python` library called `BaSC`, whose detection accuracy was tested on simulated ALMA and VLA datasets. We determined that it can find source positions with greater accuracy than current main-stream software such as `SExtractor` and `Aegean 2.0`. Also, we provided a mathematical derivation of the Bayesian resolving power formula, so as to give the limit of accuracy of the proposed method. This was then verified via numerical experiments on simulated point-source data.

The proposed approach calls for improvement in situations where the signal–to–noise ratio is low, in which case the `BayeSys` output is too complex to be clustered by the chosen

clustering algorithm. A machine learning route may be implemented in future. Further work should go toward overcoming the limitation that `BaSC` cannot cope with more than 30 sources. Generalisation of our Bayesian method from point sources to sources having arbitrary extended shape is needed, possibly with an assumption that all sources can be represented by a limited number of two-dimensional Gaussians with differing major and minor axes, corresponding to astrophysical parameters. Further research is also needed to enable `BaSC` to work on wide-field observations.

In Chapter 3, we sought a gridding function for use in both the gridding and degridding process so as to improve the accuracy of the image and visibility products. We revisited the criterion of the optimal gridding function proposed in Tan (1986), and improved the optimisation process so as to obtain gridding functions having arbitrary width up to $W = 20$. The map error function for $W = 20$ achieved a magnitude as small as 10^{-43}. Theories were revised and new implementations made so as to derive the least–misfit gridding functions. We then compared the image misfit of DFT and FFT dirty images using least–misfit functions and spheroidal functions on simulated VLA data. The least–misfit function achieves at least 100 times better image misfit level than the spheroidal function using the same W. We also compared the aliasing–suppression capability of the two functions, through numerical experiments on simulated VLA data. The results demonstrated very clearly that the least–misfit gridding function outperforms the spheroidal function using the same W, even though the spheroidal function was designed to suppress the effects of aliasing. Essentially, aliasing is part of the image misfit, so that there is no need to consider its suppression separately.

We also discussed the computational cost and the degridding process. We recommend use of the least–misfit gridding function with $W = 7$ and $x_0 = 0.25$, so as to achieve the desired single precision limit with a comparably small computational cost in both the gridding and degridding processes. If a look–up table is used, nearest–neighbour interpolation is preferred. The sampling rate should be not less than 10^6 in order to achieve the single precision limit when using the least–misfit function with $W = 7$ and $x_0 = 0.25$.

In the future, we intend to compare the accuracy of `BaSC` results from dirty images made from the same visibility data but using different gridding functions. We expect that use of the least–misfit gridding function will result in more accurate source location. Second, since the least–misfit gridding function greatly improves the accuracy of both the gridding and degridding processes, we propose to explore improvements in efficiency due to use of the least–misfit gridding function in the image reconstruction process. More numerical experiments are called for. Finally, we seek opportunities to implement the least-misfit gridding function as a gridding function option in one of the main-stream data reduction pipelines.

In Chapter 4 we proposed two wide–field imaging methods, in which the improved W–

Stacking method is based on the existing W–Stacking method. We applied our Least-misfit gridding function proposed in Chapter 3 to our improved W-Stacking method and the N-faceting method. A three-dimensional gridding in our improved W-Stacking algorithm greatly improved the accuracy in the dirty image, to 10^{-8} when the Least-misfit gridding function with $W = 8$ and $x_0 = 0.25$ was used, together with the corresponding three-dimensional correcting function. The image misfit levels achieved by the W–Projection algorithm in `CASA` and the original W–Stacking algorithm in `WSClean` are inferior by a factor of at least 10^3. In addition, since the number of w–planes required by the improved W–Stacking method is reduced compared to the original method, cutting a significant amount of FFT computational cost. As for the original W–Stacking method, if less w–planes than needed are used, the dirty images and reconstructed images produced will underestimate the fluxes of sources that are located far from the phase centre. This problem is readily overcome by increasing the image size, which indirectly increases the number of w–planes.

The N–Faceting method, on the other hand, involves imaging of multiple n–planes, where a position–independent beam is in use. The final reconstructed image is projected from the reconstructed three–dimensional image cube onto the tangent plane, followed by a convolution with the CLEAN restoring beam. We applied it to our 34–source simulated data, and displayed the dirty images and residuals by facet, as well as the reconstructed images.

We propose imminently to develop further and apply the N–Faceting method. We also hope to persuade the `WSClean` team to adopt the improved W–Stacking method as an option, along with our least–misfit gridding and correcting function pair. If `BaSC`, introduced in Chapter 2, is to be generalised to wide–field applications, the accuracy of the dirty images on the celestial sphere would be vitally important. We would like to test whether the improvement in accuracy on the dirty image increases the accuracy of source detection.

In Chapter 5, we demonstrated two applications of our improved W-Stacking method on two sets of real observation data. The improved W–stacking method successfully removes non-coplanar effects. For observations with a larger range of w, the improved W–stacking method is recommended for the generation of more accurate images at lighter computational cost, compared to the original W–stacking method.

Looking back at our original aims, we consider this thesis to have been a success, as accuracy has been greatly improved in all three projects, in differing ways. We are well aware of the general view held by radio astronomers: they seek smooth, clear and beautiful images, which are usually reconstructed images. However, it has been noted that reconstructed images are not a true representation of the sky. It was necessary to develop methods for obtaining information from dirty images, which reflect all the observational information obtained from the sky. This need underlay our development of the source extraction package `BaSC`. This

package has been proved to be more accurate than software working on CLEANed images. In view of the importance of dirty images, we made effort to optimise the gridding process so as to obtain the most accurate FFT dirty image when compared to the DFT dirty image. We proposed the least-misfit gridding and correcting function pair so as to minimise the image misfit. This gridding function is also adopted to our proposed wide-field imaging methods, which were designed to attain the best accuracy in terms of the DFT and FFT dirty image difference, even when the field of view is large.

These improvements in accuracy promise to assist scientific research into astronomical objects and phenomena in radio astronomy. We hope that the methods proposed in this thesis can be widely used in radio astronomy, so as to assist the future scientific discoveries.

APPENDIX A

As explained in §2.3.1, the choice of the parameter `Eps` has a great effect on the clustering performance of DBSCAN. This Appendix describes how we recommend a value of `Eps`.

First, we calculate the distances between every data point and its k-nearest neighbours.

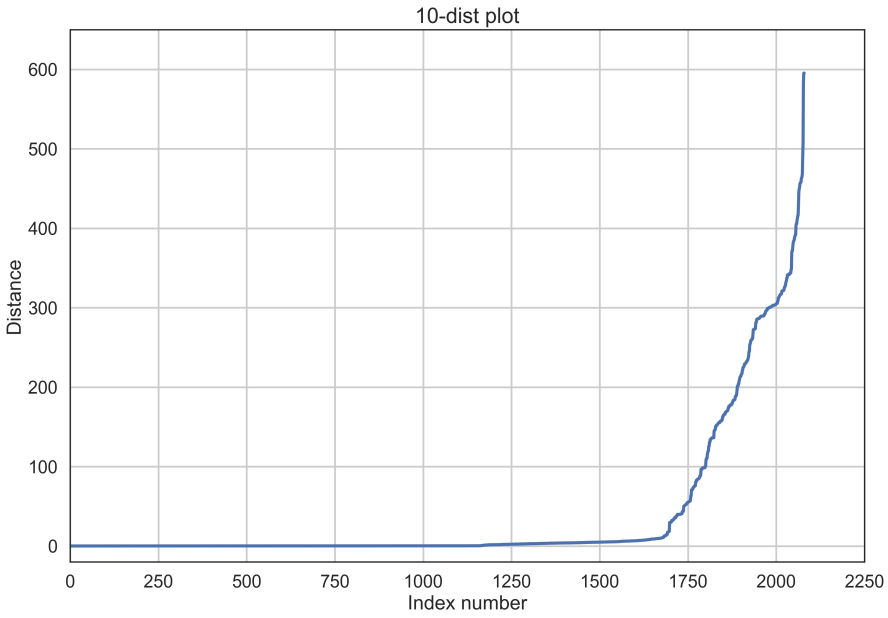


Figure A.1: Distances between every data point and its 10 nearest neighbours, sorted in ascending order.

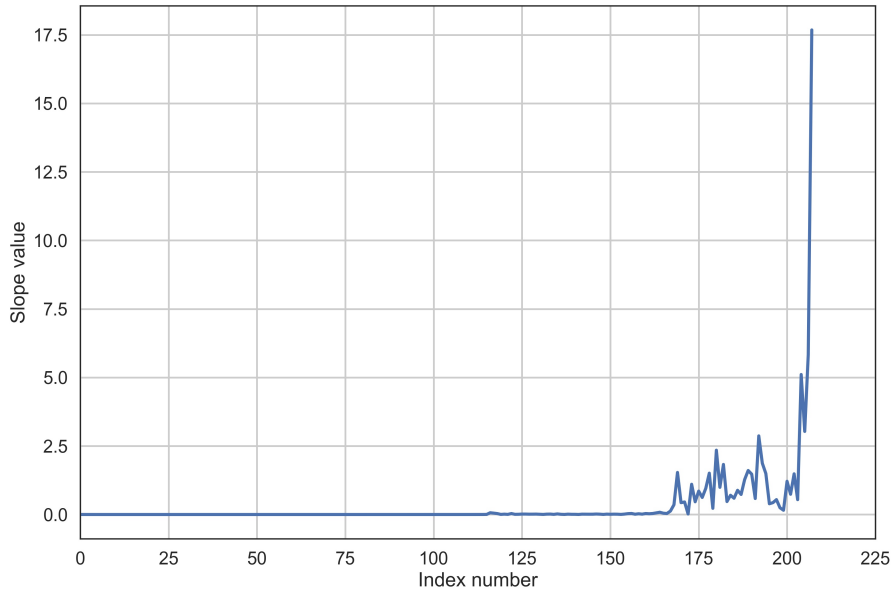


Figure A.2: The gradients of every 10 adjacent points in the 10-distribution graph.

These distances are sorted in ascending order. By default, we set $k = 10$. Figure A.1 shows a graph of the 10-distribution for the same data as used in Section 2.3.1.

The aim is to find the first valley in the k-dist graph, from which to the appropriate value of `Eps` follows. By calculating the gradients of a line comprising by every k adjacent points and plotting the gradient values, we obtain Figure A.2.

We locate the first significant leap by eye at an index number of approximately 168 in the gradient plot, but it is impossible to locate the leap by finding the maxima because there are multiple local maxima prior to the first leap. There is no evidence that it will be possible to find a threshold suitable for every single dataset.

We next determine the variances of the gradients. The first variance is set to be the variance of the first k gradients; the second is set to be the variance of the first $2k$ gradients, and so on. By this means we obtain the accumulated variance plot shown in Figure A.3. This variance plot shows clearly where the slopes change enormously, so that we may use this feature to determine the value of `Eps` for use in DBSCAN.

Based on experience, we take the last local maximum of the variance plot, and find the corresponding `Eps`. If this value is smaller than one, which is the cell size, we should assign `Eps` to one. Based on experience of processing the dataset by hand, we restrict `Eps` to be no larger than 10. This value may be changed by users after they have gained a better understanding of

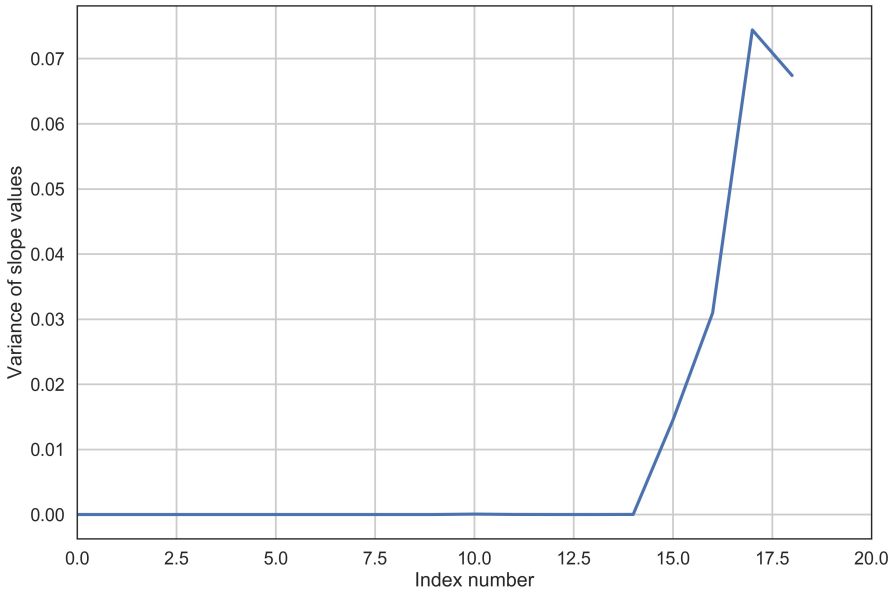


Figure A.3: The variance plot shows clearly where the gradients change enormously, so that this feature can be used to determine the value of `Eps` for use in DBSCAN.

their data.

APPENDIX B

In a M-dimensional space, the normalised Gaussian distribution can be written as

$$\mathrm{P}(\mathbf{x}) = Z^{-1} \exp\left(-\frac{1}{2}\delta\mathbf{x}^T \cdot \mathbf{A} \cdot \delta\mathbf{x}\right) \tag{B.1}$$

where $\delta\mathbf{x} \equiv \mathbf{x} - \langle\mathbf{x}\rangle$ and Z is a partition function which ensures normalisation of the probability distribution, and is defined as

$$Z \equiv \int \mathrm{d}^M\mathbf{x} \exp\left(-\frac{1}{2}\delta\mathbf{x}^T \cdot \mathbf{A} \cdot \delta\mathbf{x}\right) = (2\pi)^{M/2}(\det \boldsymbol{A})^{-1/2} \tag{B.2}$$

Upon denoting $a_{ij} = [\mathbf{A}]_{ij}$, we write the differentiation of $\log Z$ with respect to a_{ij} as

$$\frac{\delta \log Z}{\delta a_{ij}} = -\frac{1}{2}Z^{-1} \int \mathrm{d}^M\mathbf{x}\delta x_i \delta x_j \exp\left(-\frac{1}{2}\delta\mathbf{x}^T \cdot \mathbf{A} \cdot \delta\mathbf{x}\right) = -\frac{1}{2}\langle\delta x_i \delta x_j\rangle \tag{B.3}$$

At the same time, we can also write the expression for $\log Z$, based on Equation B.2, as

$$\log Z = -\frac{1}{2}\log\det\mathbf{A} + \text{constant} \tag{B.4}$$

By differentiating with respect to a_{ij}, we find that:

$$\frac{\delta \log Z}{\delta a_{ij}} = -\frac{1}{2\det\mathbf{A}}\frac{\delta\mathbf{A}}{\delta a_{ij}} \tag{B.5}$$

We use the fact that the determinant can be expressed as $\det\mathbf{A} = \sum_i a_{ij}A_{ij}$, where A_{ij} is the cofactor of a_{ij} and i, j denote indices for rows and columns in $\mathbf{A}$. We then obtain $\frac{\delta\mathbf{A}}{\delta a_{ij}} = A_{ij}$. Equation B.5 then becomes

$$\frac{\delta \log Z}{\delta a_{ij}} = -\frac{A_{ij}}{2\det\mathbf{A}} \tag{B.6}$$

Since cofactors are related to the inverse matrix by $[\mathbf{A}^{-1}]_{ij} = A_{ij}/\det \mathbf{A}$, it follows that

$$\frac{\delta \log Z}{\delta a_{ij}} = -\frac{1}{2}[\mathbf{A}^{-1}]_{ij} \tag{B.7}$$

Upon comparing Equations B.3 and B.7, we obtain

$$\langle \delta\mathbf{x}\delta\mathbf{x}^T \rangle = \mathbf{A}^{-1} \tag{B.8}$$

APPENDIX C

Table C.1 shows the location and flux of the 34 simulated point sources used in the numerical experiments in Chapters 3 and 4. Sources are scattered around the phase centre with different fluxes across the full field of view.

Sources 27, 28, 29 and 30 are at four corners of the field, which are chosen to have the

Table C.1: Locations and fluxes of 34 simulated point sources. X and Y are in pixel numbers, representing the distances from the image centre $(0, 0)$ to the corresponding point sources.

Index	X	Y	Flux (Jy)	Index	X	Y	Flux (Jy)
1	0	0	2	18	0	270	1
2	0	15	2	19	0	330	1
3	-120	180	2	20	330	0	1
4	150	-150	2	21	0	-330	1
5	300	90	2	22	-330	0	1
6	-90	300	2	23	270	270	1
7	90	-90	1	24	270	-270	1
8	-90	90	1	25	-270	270	1
9	-90	-90	1	26	-270	-270	1
10	180	90	1	27	390	390	3
11	180	180	1	28	390	-390	3
12	180	-180	1	29	-390	-390	3
13	-180	180	1	30	-390	390	3
14	-180	-180	1	31	345	0	2
15	270	0	1	32	-345	0	2
16	0	-270	1	33	0	-345	2
17	-270	0	1	34	0	345	2

largest fluxes of the 34 sources. This arrangement is chosen to test the accuracy of wide-field imaging algorithms at positions far from the phase centre.

Bibliography

Ankerst M., Breunig M. M., Kriegel H.-P., Sander J., 1999, SIGMOD Rec., 28, 49

Berkhin P., 2006, A Survey of Clustering Data Mining Techniques. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 25–71, doi:10.1007/3-540-28349-8_2, https://doi.org/10.1007/3-540-28349-8_2

Bertin E., Arnouts S., 1996, A&AS, 117, 393

Bietenholz M. F., Kassim N., Frail D. A., Perley R. A., Erickson W. C., Hajian A. R., 1997, ApJ, 490, 291

Bondi M., et al., 2003, A&A, 403, 857

Briggs D., 1995, PhD thesis, New Mexico Institute of Mining and Technology, http://www.aoc.nrao.edu/dissertations/dbriggs/

Briggs D. S., Schwab F. R., Sramek R. A., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 127

Brouw W. N., 1971, PhD thesis, Leiden Observatory, Leiden University, P.O. Box 9513, 2300 RA Leiden, The Netherlands

Brouw W. N., 1975, Methods in Computational Physics, 14, 131

Brown R. L., Wild W., Cunningham C., 2004, Advances in Space Research, 34, 555

Campello R. J. G. B., Moulavi D., Sander J., 2013, in Pei J., Tseng V. S., Cao L., Motoda H., Xu G., eds, Advances in Knowledge Discovery and Data Mining. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 160–172

Campello R. J. G. B., Moulavi D., Zimek A., Sander J., 2015, ACM Transactions on Database Systems, 10, 5:1

Clark B. G., 1980, A&A, 89, 377

Clark B. G., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 1

Cooley J. W., Tukey J. W., 1965, Mathematics of Computation, 19, 297

Cornwell T. J., 2008, IEEE Journal of Selected Topics in Signal Processing, 2, 793

Cornwell T. J., Evans K. F., 1985, A&A, 143, 77

Cornwell T., Fomalont E. B., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 187

Cornwell T. J., Perley R. A., 1992, A&A, 261, 353

Cornwell T., van Diepen G., 2008. https://www.researchgate.net/publication/242126494_Scaling_Mount_Exaflop_from_the_pathfinders_to_the_Square_Kilometre_Array

Cornwell T., Braun R., Briggs D. S., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 151

Cornwell T. J., Golap K., Bhatnagar S., 2003, Technical report, EVLA Memo 67

Cornwell T. J., Golap K., Bhatnagar S., 2005, in Shopbell P., Britton M., Ebert R., eds, Astronomical Society of the Pacific Conference Series Vol. 347, Astronomical Data Analysis Software and Systems XIV. p. 86

Cornwell T. J., Golap K., Bhatnagar S., 2008, IEEE Journal of Selected Topics in Signal Processing, 2, 647

Cornwell T. J., Voronkov M. A., Humphreys B., 2012, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. (arXiv:1207.5861), doi:10.1117/12.929336

Ellingson S. W., Clarke T. E., Cohen A., Craig J., Kassim N. E., Pihlstrom Y., Rickard L. J., Taylor G. B., 2009, IEEE Proceedings, 97, 1421

Elsmore B., Kenderdine S., Ryle Sir M., 1966, MNRAS, 134, 87

Ester M., Kriegel H.-P., Sander J., Xu X., 1996, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD'96. AAAI Press, pp 226–231

Fomalont E. B., Perley R. A., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 79

Fraley C., Raftery A. E., 1998, The Computer Journal, 41, 578

Frey B. J., Dueck D., 2007, Science, 315, 972

Gilks W., Richardson S., Spiegelhalter D., 1995, Markov Chain Monte Carlo in Practice. Chapman & Hall/CRC Interdisciplinary Statistics, Taylor & Francis

Green D. A., 2011, Bulletin of the Astronomical Society of India, 39, 289

Greisen E., 1976, VLA Scientific Memorandum No.123, National Radio Astronomy Observatory

Greisen E., 1979, VLA Scientific Memorandum No.131, National Radio Astronomy Observatory

Gull S. F., 1989, Developments in Maximum Entropy Data Analysis. Springer Netherlands, Dordrecht, pp 53–71, doi:10.1007/978-94-015-7860-8_4, https://doi.org/10.

1007/978-94-015-7860-8_4

Gull S. F., Daniell G. J., 1978, Nature, 272, 686

Gull S. F., Skilling J., 1984a, in Roberts J. A., ed., Indirect Imaging. Measurement and Processing for Indirect Imaging. p. 267

Gull S. F., Skilling J., 1984b, IEE Proceedings F - Communications, Radar and Signal Processing, 131, 646

Gull S., Skilling J., 1991, Quantified Maximum Entropy MemSys5 User's Manual. Maximum Entropy Data Consultants, http://www.mrao.cam.ac.uk/~steve/malta2009/images/MemSys5_manual.pdf

Hague P. R., Ye H., Nikolic B., Gull S. F., 2018, MNRAS,

Hall P. J., Schilizzi R. T., Dewdney P. E. F., Lazio T. J. W., 2008, URSI Radio Science Bulletin, 2008, 4

Hancock P. J., Trott C. M., Hurley-Walker N., 2018, PASA, 35

Hastings W. K., 1970, Biometrika, 57, 97

Heideman M. T., Johnson D. H., Burrus C. S., 1985, Archive for History of Exact Sciences, 34, 265

Hobson M. P., McLachlan C., 2003, MNRAS, 338, 765

Högbom J. A., 1974, A&AS, 15, 417

Hogg D. E., MacDonald G. H., Conway R. G., Wade C. M., 1969, AJ, 74, 1206

Hopkins A. M., Miller C. J., Connolly A. J., Genovese C., Nichol R. C., Wasserman L., 2002, AJ, 123, 1086

Humphreys B., Cornwell T., 2011, Square Kilometre Array Memo No. 132, 132

Jackson J. I., Meyer C. H., Nishimura D. G., Macovski A., 1991, IEEE Transactions on Medical Imaging, 10, 473

James W. Cooley J. W. T., 1965, Mathematics of Computation, 19, 297

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

Jansky K. G., 1933, Proceedings of the Institute of Radio Engineers, 21, 1387

Jansky K. G., 1935, Proceedings of the Institute of Radio Engineers, 23, 1158

Kassim N. E., Clarke T. E., Enßlin T. A., Cohen A. S., Neumann D. M., 2001, ApJ, 559, 785

Kellermann K. I., Thompson A. R., 1985, Science, 229, 123

LaRosa T. N., Kassim N. E., Lazio T. J. W., Hyman S. D., 2000, AJ, 119, 207

Landau H. J., Pollak H. O., 1961, The Bell System Technical Journal, 40, 65

Lloyd S., 1982, IEEE Transactions on Information Theory, 28, 129

Lochner M., Natarajan I., Zwart J. T. L., Smirnov O., Bassett B. A., Oozeer N., Kunz M., 2015, MNRAS, 450, 1308

Maddox S. J., Sutherland W. J., Efstathiou G., Loveday J., 1990, MNRAS, 243, 692

Mathur N. C., 1969, Radio Science, 4, 235

McInnes L., Healy J., 2017, in 2017 IEEE International Conference on Data Mining Workshops (ICDMW). pp 33–42, doi:10.1109/ICDMW.2017.12

McInnes L., Healy J., Astels S., 2017, The Journal of Open Source Software, 2

McMullin J. P., Waters B., Schiebel D., Young W., Golap K., 2007, in Shaw R. A., Hill F., Bell D. J., eds, Astronomical Society of the Pacific Conference Series Vol. 376, Astronomical Data Analysis Software and Systems XVI. p. 127

Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E., 1953, J. Chem. Phys., 21, 1087

Mohan N., Rafferty D., 2015, PyBDSF: Python Blob Detection and Source Finder, Astrophysics Source Code Library (ascl:1502.007)

Moré J. J., 1978, in Watson G. A., ed., Numerical Analysis. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 105–116

Nan R., et al., 2011, International Journal of Modern Physics D, 20, 989

Napier P. J., Thompson A. R., Ekers R. D., 1983, IEEE Proceedings, 71, 1295

Napier P. J., Bagri D. S., Clark B. G., Rogers A. E. E., Romney J. D., Thompson A. R., Walker R. C., 1994, IEEE Proceedings, 82, 658

Narayan R., Nityananda R., 1986, ARA&A, 24, 127

Nord M. E., Lazio T. J. W., Kassim N. E., Hyman S. D., LaRosa T. N., Brogan C. L., Duric N., 2004, AJ, 128, 1646

O'Sullivan J. D., 1985, IEEE Transactions on Medical Imaging, 4, 200

Offringa A. R., et al., 2014, MNRAS, 444, 606

Pedregosa F., et al., 2011, Journal of Machine Learning Research, 12, 2825

Perley R. A., 1989, in Perley R. A., Schwab F. R., Bridle A. H., eds, Astronomical Society of the Pacific Conference Series Vol. 6, Synthesis Imaging in Radio Astronomy. p. 259

Perley R. A., Cornwell T. J., 1991, in Cornwell T. J., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 19, IAU Colloq. 131: Radio Interferometry. Theory, Techniques, and Applications. pp 184–187

Reber G., 1940, Proceedings of the IRE, 28, 68

Romein J. W., 2012, in Proceedings of the 26th ACM International Conference on Supercomputing. ICS '12. ACM, New York, NY, USA, pp 321–330, doi:10.1145/2304576.2304620

Ryle M., Neville A. C., 1962, MNRAS, 125, 39

Ryle M., Vonberg D. D., Bragg W. L., 1948, Proceedings of the Royal Society of London Series A, 193, 98

Salak D., Nakai N., Hatakeyama T., Miyamoto Y., 2016, ApJ, 823, 68

Sault R. J., Teuben P. J., Wright M. C. H., 1995, in Shaw R. A., Payne H. E., Hayes J. J. E., eds,

Astronomical Society of the Pacific Conference Series Vol. 77, Astronomical Data Analysis Software and Systems IV. p. 433 (arXiv:astro-ph/0612759)

Sault R. J., Bock D. C.-J., Duncan A. R., 1999, A&AS, 139, 387

Savage R. S., Oliver S., 2007, ApJ, 661, 1339

Schilizzi R. T., 2004, in Oschmann Jr. J. M., ed., Proc. SPIEVol. 5489, Ground-based Telescopes. pp 62–71, doi:10.1117/12.551206

Schomberg H., Timmer J., 1995, IEEE Transactions on Medical Imaging, 14, 596

Schubert E., Sander J., Ester M., Kriegel H. P., Xu X., 2017, ACM Transactions on Database Systems, 42, 19:1

Schwab F., 1978, VLA Scientific Memorandum No.129, National Radio Astronomy Observatory

Schwab F. R., 1980, VLA Scientific Memorandum No.132, National Radio Astronomy Observatory

Schwab F. R., 1981, VLA Scientific Memorandum No.156, National Radio Astronomy Observatory

Schwab F. R., 1984, in Roberts J. A., ed., Indirect Imaging. Measurement and Processing for Indirect Imaging. pp 333–346

Schwab F. R., Cotton W. D., 1983, AJ, 88, 688

Sivia D. S., Skilling J., 2006, Data analysis: a Bayesian tutorial. Oxford science publications, Oxford University Press

Skilling J., 1989, Classic Maximum Entropy. Springer Netherlands, Dordrecht, pp 45–52, doi:10.1007/978-94-015-7860-8_3, https://doi.org/10.1007/978-94-015-7860-8_3

Skilling J., 2004, BayeSys manual, http://www.mrao.cam.ac.uk/~steve/algor2008/images/BayeSys_manual.pdf

Slepian D., Pollak H. O., 1961, The Bell System Technical Journal, 40, 43

Smith D. M. P., Young A., Davidson D. B., 2017, A&A, 603, A40

Stratton J. A., 1935, Proceedings of the National Academy of Science, 21, 51

Sullivan III W. T., 1991, in Cornwell T. J., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 19, IAU Colloq. 131: Radio Interferometry. Theory, Techniques, and Applications. p. 132

Swarup G., Ananthakrishnan S., Kapahi V. K., Rao A. P., Subrahmanya C. R., Kulkarni V. K., 1991, Current Science, Vol. 60, NO.2/JAN25, P. 95, 1991, 60, 95

Tan S. M., 1986, PhD thesis, University of Cambridge

Thompson A. R., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 11

Thompson A. R., Bracewell R. N., 1974, AJ, 79, 11

Thompson A. R., Clark B. G., Wade C. M., Napier P. J., 1980, ApJS, 44, 151

Thompson A. R., Moran J. M., Swenson Jr. G. W., 2017, Interferometry and Synthesis in Radio Astronomy, 3rd Edition, doi:10.1007/978-3-319-44431-4.

Tingay S. J., et al., 2013, PASA, 30, e007

Trakhtenbrot B., Lira P., Netzer H., Cicone C., Maiolino R., Shemmer O., 2017, ApJ, 836, 8

Varbanescu A., van Amesfoort A., Sips H. J., Cornwell T., Elmegreen B., Mattingly A., van Diepen G., van Nieuwpoort R., 2018

Venkata U. R., 2010, PhD thesis, New Mexico Institute of Mining and Technology

Walker R. C., 1999, in Taylor G. B., Carilli C. L., Perley R. A., eds, Astronomical Society of the Pacific Conference Series Vol. 180, Synthesis Imaging in Radio Astronomy II. p. 433

Wells D. C., 1985, in di Gesu V., Scarsi L., Crane P., Friedman J. H., Levialdi S., eds, Data Analysis in Astronomy. p. 195

Whittam I. H., Green D. A., Jarvis M. J., Riley J. M., 2017, MNRAS, 464, 3357

Yashar M., Kemball A., 2009, TDP Calibration and Processing Group (CPG) Memo 3

de Vaate J. G. B., et al., 2011, in 2011 XXXth URSI General Assembly and Scientific Symposium. pp 1–4, doi:10.1109/URSIGASS.2011.6051197

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