

# P8 Exploring new Architectures for Compressed Sensing Reconstructions in the MeerKAT era

Jonas Schwammberger

January 31, 2019

## Abstract

The new MeerKAT Radio Interferometer poses an image reconstruction problem on a large scale. It measures an incomplete set of Fourier components, which have to be reconstructed by an imaging algorithm. Compressed Sensing reconstructions have the potential to improve the effective accuracy of MeerKAT, but so far have higher runtime costs compared to state-of-the-art CLEAN implementation. Both Compressed Sensing and CLEAN reconstructions use the non-uniform FFT approximation to cycle between Fourier and image space. But compared to CLEAN, Compressed Sensing algorithms need more cycles to converge, which is one of the reasons why they have higher runtime costs.

In this project, we investigate if Compressed Sensing algorithms can reduce the costs by replacing the non-uniform FFT approximation. We discuss three alternatives and create a new algorithm which does not need the non-uniform FFT during optimization. We leveraged the starlet transform and created a Compressed Sensing algorithm, which only needs to calculate the transform for non-zero basis functions. This allows us to use the direct Fourier Transform, which naturally extends to wide field of view measurements without any approximations. We demonstrate superior image quality on simulated data and extrapolate the runtime costs to a real-world MeerKAT observation. Sadly, our algorithm was unable to reduce the runtime costs of large scale reconstructions.

Currently, the non-uniform FFT seems an efficient way to approximate the Fourier Transform, which leads to iterative reconstruction algorithms. The number of measurements continue to rise in near future. At some point, distributed computing will become necessary for large scale reconstructions. The iterative nature of the non-uniform FFT becomes an issue for distributed computing. It is still an open question how to create a distributed large scale reconstruction algorithm. As it is, our approach is too expensive. But with future approximations, we may become competitive.

## Contents

<b>1 Compressed Sensing Image Reconstruction for MeerKAT</b>	<b>1</b>
1.1 The basic reconstruction problem in Radio Interferometry . . . . .	1
1.2 The non-uniform FFT and the Major Cycle . . . . .	2
<b>2 Challenges for imaging MeerKAT data</b>	<b>4</b>
2.1 Wide Field of View Imaging and the third Fourier dimension . . . . .	4
2.1.1 State of the art: $w$ -stacking algorithm . . . . .	5
2.2 Basic Calibration and Self-Calibration . . . . .	5
<b>3 Alternatives to the non-uniform FFT</b>	<b>7</b>
3.1 Optimal projection on a uniform grid . . . . .	7
3.2 Spherical Harmonics . . . . .	8
3.3 Direct Fourier Transform . . . . .	8
<b>4 Compressed Sensing with Coordinate Descent and the direct Fourier Transform</b>	<b>10</b>
4.1 The Starlet Transform . . . . .	11
4.1.1 Non-negative Starlet Modification . . . . .	13
4.2 Proof of concept Implementation . . . . .	13
4.2.1 Iteration Scheme and Heuristics . . . . .	14
<b>5 Test on simulated data</b>	<b>15</b>
5.1 Super-resolution of two point sources . . . . .	15
5.2 Super resolution of mixed sources . . . . .	16
<b>6 Runtime cost comparison on a real-world MeerKAT reconstruction</b>	<b>19</b>
6.1 Cost function of an idealized Coordinate Descent . . . . .	19
6.2 Cost function of WSCLEAN . . . . .	20
6.3 Comparison on a MeerKAT reconstruction problem . . . . .	20
6.4 Approximations as key for going large scale . . . . .	22
<b>7 Compressed Sensing reconstructions in the MeerKAT era</b>	<b>23</b>
<b>8 Ehrlichkeitserklärung</b>	<b>27</b>

# 1 Compressed Sensing Image Reconstruction for MeerKAT

Image reconstruction problems arise in a variety of fields. Noise from different sources like external interference, loss of samples, or the instrument itself, all introduce corruptions that a reconstruction algorithm should remove. In a sense, we need a model for what the truly observed image can look like and filter out the structures due to noise. A reconstruction algorithm needs a prior for what the image is.

This has led to the theory of Compressed Sensing[1, 2], which allows us to model the reconstruction as a convex optimization problem. It gives us a theoretical framework for the analysis of reconstruction algorithms. Compressed Sensing gives us theoretical guarantees that, under the right prior, the algorithm is virtually guaranteed to reconstruct the observed image, potentially above the accuracy limit of the instrument.

In this work we apply the theory of Compressed Sensing to the reconstruction problem of Radio Interferometers. More specifically, we apply it on the MeerKAT interferometer, which poses the problem on a new scale of data volume. New Interferometers measure billions of Fourier components, which get reconstructed to a single image. The raw measurements easily take up hundreds of gigabytes of space. The current state of the art reconstruction is based on the CLEAN[3, 4] algorithm. Although newer Compressed Sensing reconstructions showed superior image quality[5, 6], they have higher runtime costs than CLEAN.

CLEAN and current Compressed Sensing algorithms use the non-uniform FFT approximation[7, 8] to cycle between measurements and image space. Compressed Sensing algorithms need more cycles to converge to an image, which is one source of the higher runtime costs. In this project we investigate alternatives to the non-uniform FFT for Radio Interferometers. We created a proof-of-concept algorithm which does not need the non-uniform FFT during optimization. Instead, we use the direct Fourier Transform which potentially leads to a distributable Compressed Sensing algorithm. Sadly, our new algorithm cannot reduce the runtime costs compared to CLEAN on large scale reconstructions.

New Interferometers pose the reconstruction problem on an ever larger scale. In the near future distributed computing will become necessary for large scale reconstructions. The cyclic nature of the non-uniform FFT currently does not lend itself to distribution on a large scale. Distributing the image reconstruction is still an open problem.

The rest of this article is structured as the following: We begin by introducing the basic image reconstruction problem in radio interferometry. We show the Major Cycle architecture for reconstruction algorithms, which is derived from the non-uniform FFT. Section 2 introduces the difficulties of imaging MeerKAT data. We move towards competing architectures in section 3. In the following sections 4, 5 and 6 we introduce our Compressed Sensing algorithm, show the reconstruction quality on simulated MeerKAT data and extrapolate the runtime costs on a real world MeerKAT measurement.

## 1.1 The basic reconstruction problem in Radio Interferometry

We start with a simplified measurement equation for Radio Interferometers, (1.1). An interferometer measures an incomplete set of Fourier Components  $V$  (called Visibilities in Radio Astronomy) from the sky image  $I$  at position  $x$  and  $y$ . We want to reconstruct the image  $I$ , while the instrument measures  $V$  in a noisy environment. Since the term  $e^{2\pi i(ux+vy)}$  is the Fourier Transform, the image can be calculated from the Visibilities by the inverse Fourier Transform.

$$V(u, v) = \int \int I(x, y) e^{2\pi i(ux+vy)} dx dy \quad (1.1)$$

However, the Visibilities are incomplete and noisy. The inverse Fourier Transform does not give us the observed image, but a corrupted "dirty" image. It introduces structures in the dirty image which were not ob-

served. The incomplete Visibility coverage effectively convolves the observed image with a Point Spread Function. A reconstruction algorithm has to decide which structures were truly observed, and which are due to noise and incomplete Visibility coverage. In the framework of Compressed Sensing, this leads us to a minimization problem with the following objective function:

$$\underset{x}{\text{minimize}} \quad \|V - Fx\|_2^2 + \lambda \|P(x)\|_1 \quad (1.2)$$

The objective (1.2) is split into two terms, the data and regularization term. The data term forces the reconstructed image to be as close to the measurements as possible, while the regularization term penalises unlikely images. Overall, we try to find the optimal image, which is as close to the Visibilities as possible, but also has the smallest regularization penalty according to some prior function  $P()$ . The parameter  $\lambda$  weights the trade-off between the data and regularization term. The theory of Compressed Sensing states that if our prior function  $P()$  models the observed image well, we are virtually guaranteed to find it at the minimum of our objective function (1.2).

The question is, what is a good prior? This depends on the image content. For example if our Interferometer measures an image containing only stars (called point-sources), our observed image will only contain a non-zero pixels at the locations of the stars. In this case, the  $L1$  norm<sup>1</sup> is a good prior. It will force the reconstruction to have as few non-zero pixels as possible, and we are likely to find the truly observed image at the minimum of (1.2).

In theoretical terms, the reconstruction problem for Radio Interferometry boils down to finding a good prior function  $P()$ , and an efficient optimization algorithm to minimize (1.2). In the real-world, large scale reconstructions create additional problem. For MeerKAT, we have millions of pixels and several billions of Visibilities in a single reconstruction. The Fourier Transform Matrix  $F$  with the size of Visibilities times pixels becomes expensive to either compute or keep in memory. Also, we cannot use the Fast Fourier Transform, because the interferometer measures an incomplete and non-uniformly sampled set of Visibilities.

The large Fourier Transform matrix is common for image reconstructions in Radio Astronomy. MeerKAT just poses the same problem on an even larger scale. The state-of-the-art way of dealing with  $F$  is to use the non-uniform FFT approximation.

## 1.2 The non-uniform FFT and the Major Cycle

We look at how the non-uniform FFT leads to the Major Cycle architecture for CLEAN reconstructions. We discuss how the Major Cycle improves CLEAN reconstructions and show how Compressed Sensing algorithms use the non-uniform FFT to benefit in a similar manner.

$$\underset{x}{\text{minimize}} \quad \|I_{\text{Dirty}} - x \star PSF\|_2^2 + \lambda \|x\|_1 \quad , \quad I_{\text{Dirty}} = \hat{F}^{-1}V \quad (1.3)$$

A CLEAN image reconstruction in the Major Cycle consists of two different operations: A non-uniform FFT, and a CLEAN deconvolution. CLEAN's minimization objective can be approximately formalized in (1.3), where  $\hat{F}$  is the non-uniform FFT approximation. The non-uniform FFT approximates the inverse Fourier Transform and calculates the dirty image. The CLEAN deconvolution uses the Point Spread Function, deconvolves the dirty image and approximates the observed image. To increase the overall accuracy of the reconstruction, CLEAN is used inside the Major Cycle architecture depicted in figure 1. In each Cycle, the residual Visibilities get transformed by the non-uniform FFT, CLEAN approximates the deconvolution, and the new residual image gets transformed back to Visibilities. Over several Major Cycles the error from the non-uniform FFT and from

---

<sup>1</sup>simply the absolute value of the image,  $abs(x)$

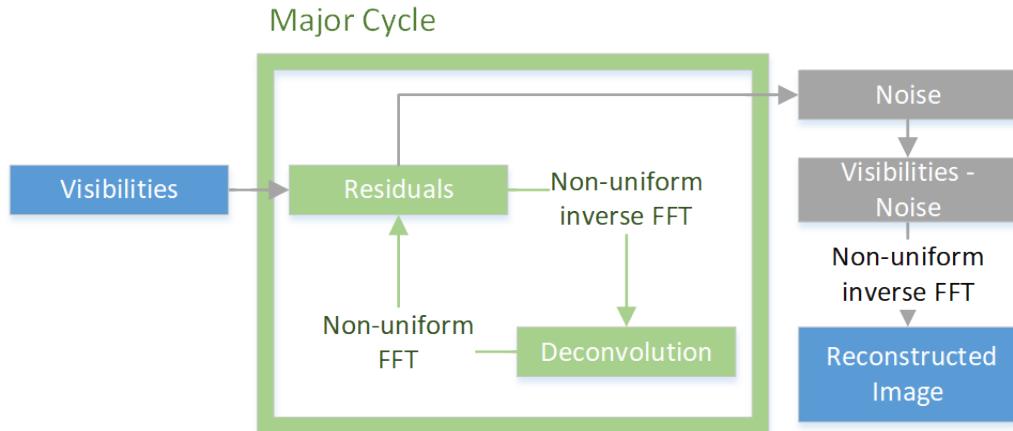


Figure 1: The Major Cycle Framework

the CLEAN deconvolutions get minimized. In a sense, the Major Cycle is a simple optimization algorithm for two different errors.

In reality, CLEAN uses a number of heuristics which are hard to formalize. It minimizes roughly the objective (1.3). In that sense, CLEAN is similar to a Compressed Sensing reconstruction with the L1 as regularization and a non-uniform FFT as approximation.

Current Compressed Sensing reconstructions also use the non-uniform FFT and cycle between Visibility and image space[5, 6], but do not use a deconvolution. Instead, they use the image to calculate the regularization penalty. But the effects are similar. Compressed Sensing minimizes the error from the non-uniform FFT approximation and the error from incomplete measurements over several cycles of non-uniform FFT approximations. As we will see in section 2, wide field of view imaging also works identical any algorithm using the non-uniform FFT. For our intents and purposes, Compressed Sensing reconstructions with the non-uniform FFT use the Major Cycle architecture. In this project, we try to find a new architecture optimized for Compressed Sensing algorithms.

## 2 Challenges for imaging MeerKAT data

We introduced the basic measurement equation and the imaging problem for Radio Interferometers. MeerKAT introduces a variety of new challenges for image reconstruction. In this work, we limit the scope to the third Fourier term of wide field of view imaging, and discuss the basics of calibration. Both directly affect how the reconstruction algorithm deals with the Fourier Transform and the wider architecture of the reconstruction. In this section, we look at the wide field of view measurement equation, discuss the basics of calibration and how they are solved in the Major Cycle architecture.

### 2.1 Wide Field of View Imaging and the third Fourier dimension

In wide field of view imaging, the simplifications we could make from the basic measurement equation (1.1) do not hold. The Visibility space of an interferometer actually has a third  $w$ -term. This leads us to the wide field of view measurement equation (2.1).

$$V(u, v, w) = \int \int \frac{I(x, y)}{\sqrt{1 - x^2 - y^2}} e^{2\pi i [ux + vy + w(\sqrt{1-x^2-y^2} - 1)]} dx dy \quad (2.1)$$

For a small field of view, the term  $\sqrt{1 - x^2 - y^2} \approx 1$ , which simplifies to our original (1.1), we can ignore the  $w$ -term and use the two dimensional Fourier transform. Older interferometers typically created small field of view observations, where the basic measurement equation was accurate enough. But MeerKAT produces wide field of view observations. The  $w$ -term cannot be ignored anymore and has to be corrected. The figure 2b shows the effect of ignoring the  $w$ -term for wide-field of view observations. The image gets more distorted away from the center. Emissions at the edges of the image get "torn" apart.



Figure 2: Celestial sphere distortion on simulated data. Source: [9]

Let us look at the wide field of view measurement equation (2.1) in detail and explain this phenomenon. We see that the image  $I(x, y)$  is still two dimensional, even with a three dimensions in Visibility space. The observed image of the interferometer is not on a flat plane, but on a curved surface(hence the  $w$ -term)[10]. More precisely, the instrument looks at the inside wall of the celestial sphere. The two dimensional Fourier

transform approximates the curved surface with a flat plane, where the tangent point is typically the image center. The further away we move from the tangent point, the more distortion gets added by the curvature, represented by the term  $\sqrt{1 - x^2 - y^2}$ . The curve adds a phase shift the further away we move from the tangent point. If the reconstruction algorithm ignores the  $w$ -component for a wide field of view, the phase-shift gets severe enough to decorrelate the Visibilities, "tearing" apart the image structures of (2).

The effect of the  $w$ -term is one example of a Direction Dependent Effect (DDE). Wide field of view imaging introduces a number of DDE's, like distortions from the ionosphere and bleed over from different polarization. Correcting for the DDE in the Major Cycle architecture is under research[11], and is therefore out of scope for this project. We only correct for the  $w$ -term which directly influences the Fourier Transform and the architecture of reconstruction algorithms.

### 2.1.1 State of the art: $w$ -stacking algorithm

The third Fourier breaks the two dimensional relationship between image and Visibilities, which keeps us from using the non-uniform FFT in the Major Cycle architecture. Before the  $w$ -stacking algorithm was developed, a wide field of view image was faceted into image patches where the  $w$ -term could be approximated with the small field of view measurement equation.  $w$ -stacking is the state-of-the-art way of approximating for the  $w$ -correction in an efficient manner. It calculates a  $w$ -corrected dirty image from three dimensional Visibilities.

The  $w$ -stacking algorithm, shown in figure 3 replaces the non-uniform FFT. It groups the Visibilities with similar  $w$ -term into the same layer. Multiple layers are then stacked and transformed independently with the non-uniform FFT. For each stack, the  $w$ -correction is performed for the center  $w$ -component. The last step is summing over all stacked images, giving us the final dirty image. From here on a standard CLEAN deconvolution, or a Compressed Sensing algorithm can search for the observed image.



Figure 3: The Major Cycle Framework

Afterwards, the whole process is reversed. The image gets copied back into the stacks, the  $w$ -correction gets reversed and the non-uniform FFT calculates the Visibilities per stack.

If we choose uses as many stacks as Visibilities, we end up with an exact  $w$ -corrections. In practice, many Visibilities have a similar  $w$ -term.  $w$ -stacking approximates the correction and allows the Major Cycle to distribute the forward- and backwards non-uniform FFT to a certain extend.

## 2.2 Basic Calibration and Self-Calibration

The Visibilities measured by a Radio interferometer require calibration before an image can be reconstructed. Calibration corrects the amplitude and phase of the measured Visibilities for instrumental differences and direction independent effects. The basic calibration scheme uses a second observation where the true image was known, and solved for the missing calibration parameters  $C$ , shown in equation (2.2).  $V_2$  are the Visibilities from the calibration observation, and  $x_2$  is the known image. The resulting calibration parameters are then used in the observation of interest.

$$\underset{C}{\text{minimize}} \quad \|CV_2 - F(x_2 \star PSF)\|_2^2 \quad (2.2)$$

Accurate calibration is necessary for a high-fidelity image reconstruction. For small field of view observations, the number of calibration parameters is relatively small and if the observation spans a short period of time, sufficiently accurate. For observations over a longer time period, the calibration parameters drift significantly, and wide field of view increases the number of necessary parameters.

The equation (2.2) is essentially the reverse operation of image reconstruction. If we know at least parts of the observed image, we can improve the calibration parameters. This has led to the self-calibration scheme where only an initial calibration is done with a secondary observation, and the imaging algorithm is expected to improve upon the calibration parameters. In self-calibration mode, the imaging algorithm creates a reconstruction from initial calibration. The reconstruction is used to improve the calibration, which in turn gets used for the next reconstruction. The imaging algorithm iteratively solves for both, the observed image  $x$  and the calibration parameters  $C$ .

For wide field of view, observations over a long time period, self-calibration leads to a more accurate reconstruction than otherwise possible. As we will see in the next section, self-calibration has implication on the Fourier Transform of the reconstruction algorithm and therefore on the overall architecture.

### 3 Alternatives to the non-uniform FFT

The MeerKAT creates wide field of View observations. For accurate image reconstruction, the algorithm also has to optimize the calibration parameters  $C$  together with the observed image  $x$ . This leads us to the following optimization objective (3.1) for MeerKAT imaging.  $F_{wof}$  represents the wide field of view Fourier Transform with three dimensional Visibilities and two dimensional image.

$$\underset{C,x}{\text{minimize}} \ \|CV - F_{wof}x\|_2^2 + \lambda \|P(x)\|_1 \quad (3.1)$$

State of the art CLEAN algorithms use the Major Cycle architecture for image reconstruction. They use the  $w$ -stacking algorithm to approximate  $F_{wof}$  and cycle between Visibility and image space. Current Compressed Sensing approaches use the  $w$ -stacking algorithm in a similar manner[5, 6, 10, 12], but need more cycles to converge to an image. This is one reason for the overall higher runtime costs of Compressed Sensing. Various research focuses on improving the reconstructions in the context of the non-uniform FFT and  $w$ -stacking. Pratley et al[8] optimized the accuracy of the non-uniform FFT for Compressed Sensing reconstructions. Dabbech et al.[13] investigated  $w$ -term approximations for reducing runtime costs.

Alternatives to the non-uniform FFT are rarely discussed in Radio Astronomy. We found only Hardy[19] which investigated the direct Fourier Transform for Compressed Sensing reconstructions. In this project, we investigate if we can use an alternative to the non-uniform FFT and reduce the runtime costs of Compressed Sensing. We discuss three alternatives: Optimal projection on a uniform grid, Spherical Harmonics and the direct Fourier Transform.

#### 3.1 Optimal projection on a uniform grid

The  $w$ -stacking algorithm continually cycles between non-uniform Visibility space and uniform image space. Over several cycles a reconstruction algorithm finds the optimal projection on the uniform grid together with the observed image. Here we discuss the potential of searching for the optimal projection first and reconstruct the image in a second step.

$$\underset{I_{dirty}}{\text{minimize}} \ \|V - \tilde{F}_{wof}I_{dirty}\|_2^2 \quad (3.2)$$

$$\underset{x}{\text{minimize}} \ \|I_{dirty} - x * PSF\|_2^2 + \lambda \|P(x)\|_1 \quad (3.3)$$

The first objective (3.2) solves for the optimal projection on a uniform grid. It uses the  $w$ -stacking algorithm, represented as  $\tilde{F}_{wof}$ , which approximates the wide field of view Fourier transform. After we found the optimal projection, we do not need the original Measurements for further processing. The objective (3.3) only needs the dirty image and the Point Spread Function to find the observed image according to some prior  $P()$ . In this setting, Compressed Sensing reconstructions and CLEAN need the same amount of  $w$ -stacking cycles. (3.2).

The main advantage for this architecture is that it reduces the problem to a uniformly-sampled grid, and ignore the original Visibilities in later steps. However, note that (3.2) and (3.3) do not solve for calibration parameters.  $C$  is defined in the non-uniformly sampled space. Unless we find a way to project the calibration problem on a uniform grid too, we cannot ignore the original Visibilities and the main advantage of this setting vanishes.

In practice, there is a second difficulty with the optimal projection: The  $PSF$  of objective (3.3) is difficult to represent accurately on a uniform grid. It varies for each pixel, the  $PSF$  at the center of the dirty image is not identical to the  $PSF$  for edge pixels. For the most accurate result, we would need a  $PSF$  for each pixel. The

naive way to calculate a  $PSF$  for a pixel is to set all amplitudes of our Visibilities to 1, shift the phase to the desired pixel, and calculate the Fourier Transform. The naive way to calculate a  $PSF$  for each pixel leads to a quadratic runtime.

CLEAN gets away with a constant  $PSF^2$ , because of the Major Cycle architecture. In each cycle, it reduces the error it introduced by the constant  $PSF$  of the previous cycle, leading to a more accurate reconstruction. When we project on a uniformly sampled grid, we need several  $PSF$ 's for a comparable accuracy to the Major Cycle, and this number dictates whether we can reduce the runtime costs. The question, how many  $PSF$ 's are necessary, seems currently unknown in the Radio Astronomy community, as it becomes irrelevant in the Major Cycle architecture.

Solving for the Calibration parameters  $C$  of our reconstruction problem (3.1) destroys the main advantage of the projection.

### 3.2 Spherical Harmonics

Spherical Harmonics are a different way to represent the measurements of an interferometer. Carozzi[14] derived a measurement equation based on the fact that the Visibilities fulfils the Helmholtz equation. In practical terms, Carozzi showed we can replace  $F_{wof}$  in our reconstruction problem (3.1) the Spherical Harmonics Transform.

Compared to the Fourier domain, Spherical Harmonics naturally represent wide field of view interferometers and do not have a third dimension. This opens up new designs for Compressed Sensing reconstructions. For example, we can reconstruct the image by in-painting the missing Spherical Harmonics. Because they naturally represent curved surfaces, we can in-paint in a two dimensional space instead of three. Another direction is to image on the sphere directly. MCewen et al.[16] showed a way to put two dimensional wavelets on the sphere. A Compressed Sensing reconstruction can use a spherical wavelet as regularization and may simplify the transformation between measurements and reconstruction space.

Sadly, there is little published research in this area for Radio Astronomy image reconstructions. MCewen et al.[16] improved the runtime costs of simulations with the spherical Haar wavelet. Later work[10] showed a proof-of-concept Compressed Sensing reconstruction projected on the sphere, which improved the image quality. However, it did not use Spherical Harmonics and [10] is still based on the Fourier transform. At this point we have not found a proof-of-concept reconstruction algorithm which uses Spherical Harmonics to reduce runtime costs.

There seems untapped potential for new, cheaper reconstruction algorithms with Spherical Harmonics. But replacing the  $F_{wof}$  with the Spherical Harmonics Transform might have wide-ranging consequences. The Fourier relationship, self-calibration, the effect of the ionosphere, everything discussed in section 2 is based on a plane wave arriving at the instrument[17, 18]. Spherical Harmonics are not. They are derived from a different property of the signal. If we use the Spherical Harmonics Transform, we do not know if we can still solve for the calibration parameters  $C$  in (3.1). We may need to re-invent self-calibration, ionosphere distortion and more for Spherical Harmonic reconstructions.

### 3.3 Direct Fourier Transform

The direct Fourier Transform replaces the  $w$ -stacking algorithm with the full, explicit Fourier Matrix  $F_{wof}$ . Hardy[19] showed the direct Fourier Transform leads to a simplified reconstruction algorithm which simplifies parallel computing. In this setting, we can calculate the exact wide field of view Fourier Transform from image

---

<sup>2</sup>CLEAN implementations in Radio Astronomy typically use a constant  $PSF$ . This is not necessary, one can also implement CLEAN with a varying  $PSF$ .

to Visibility space. For self-calibration, we require a transformation from image to Visibility space which the direct Fourier Transform provides. The downside is of course we need the whole matrix  $F_{wof}$  either in memory, or need to re-calculate it on the fly. For MeerKAT reconstructions, where we reconstruct millions of pixels from several billions of Visibilities, the matrix becomes too large for any practical application.

However, the full matrix is not needed. Remember that CLEAN assumes the image is sparse, meaning only a few pixels are non-zero. In practice, even with extended emissions in the image, large regions of pixels are zero. They do not contribute to the reconstruction. We only need to calculate the Transform for non-zero components. With Compressed Sensing, we are not required to reconstruct in the image space. We can use a space where the image can be represented in even fewer non-zero components, reducing the memory requirement and runtime costs of the direct Fourier Transform. In this project, we use starlets as sparse image representation. We investigate if this approach is enough to let the direct Fourier Transform scale to MeerKATs data volume.

## 4 Compressed Sensing with Coordinate Descent and the direct Fourier Transform

Until we extrapolate the runtime costs of the direct Fourier Transform in section 6, we ignore both the  $w$ -term and calibration for simplicity. The direct Fourier Transform works identical for wide field of view observations. Self-calibration and Coordinate Descent both require a transformation from image to Visibility space. By using Coordinate Descent, we do not need to handle self-calibration explicitly. We show the principle of our approach at a simplified Coordinate Descent example, which leads us to the following objective function (4.1).

$$\underset{x}{\text{minimize}} \quad \|V - Fx\|_2^2 + \lambda \|x\|_1 \quad (4.1)$$

We can use a number of different optimization algorithms to optimize (4.1). Note however in the one dimensional case, where we only have one pixel, the data term of the objective (4.1) forms a parabola and the regularization term a shrink operation<sup>3</sup>. The optimum for a single pixel can be calculated by solving for the minimum of the parabola first, followed by a shrink operation. With Coordinate Descent we can exploit this property. We fix all pixels except for one and iteratively solve for the current minimum of each pixel. If the pixels would be independent of each other, then our algorithm would converge within one full iteration. In general this is not the case for Radio Interferometric image reconstructions, and we need several iterations over all pixels converge. The full Coordinate Descent algorithm can be implemented in a few lines of python code:

```

1 def coordinate_descent(V_residual, x, lambda, max_iter):
2     for k in range(0, max_iter):
3         for i in pixels_row:
4             for j in pixels_column:
5                 x_old = x[i, j]
6                 fourier_column = calculate_fourier_transform(i, j)
7                 fr = real(fourier_column)
8                 fi = imag(fourier_column)
9                 rr = real(V_residual)
10                ri = imag(V_residual)
11
12                #find apex
13                a = sum(fr**2 + 2*fr*fi + fi**2)
14                b = sum(fr*rr + fr*ri + fi*rr + fi*ri)
15                x_new = b / a + x_old
16
17                x_new = shrink(x_new, lambda)
18                x[i, j] = x_new
19                V_residual = V_residual - fourier_column * (x_new - x_old)

```

---

Coordinate Descent has to iterate over every pixels possibly several times. How quickly Coordinate Descent converges in theory is not well understood. Our optimization function (4.1) falls in the class of quadratic programming with a strictly convex objective. In this case, Coordinate Descent is guaranteed to converge at least linearly[20]. Sadly, real-world objective functions for image reconstruction are often not strictly convex. Usually our image is constrained to have only positive pixels[10], which breaks the strictly convex property of the objective function. In our environment, the convergence guarantees of Coordinate Descent are not well understood in theory.

<sup>3</sup>The shrink operation reduces the magnitude of the pixel by  $\lambda$ . For example: Pixel  $x = -12.4$ ,  $\lambda = 0.6$ . The new pixel value after shrinkage follows as  $\text{shrink}(-12.4, 0.6) = -11.9$

In practice, Coordinate Descent is used together with heuristics to speed up convergence. In our simplified example, we can leverage the fact that only a small number of pixels will be non-zero. The simple Coordinate Descent algorithm wastes resources on mostly zero pixels. A better scheme would be the active set heuristic[21]: We iterate over every pixel once. Then, for a given number of iterations, we only check the pixels that have changed. This is an improvement, but too expensive in the context of MeerKAT. A reconstruction problem can have billions of Visibilities and several millions of pixels. Since  $F$  has the size of Visibilities times pixels, it gets too expensive to calculate all columns even once.

However, it turns out we do not need to if we imitate CLEAN: For CLEAN, the non-uniform FFT approximates the 'dirty' image, which is corrupted by the effects of incomplete measurements. CLEAN then subtracts a fraction of the PSF at the largest pixel value. In other words, the largest pixel value in the dirty image is the most likely non-zero pixel. We can use the dirty image approximation for a 'probability distribution' of non-zero pixels. It does not need to be accurate, since the actual reconstruction is done with the direct Fourier Transform.

This is, in principle, how the proof-of-concept algorithm works. Instead of reconstructing an image, it reconstructs in the starlet space. It uses the starlet transform of the dirty image to find likely non-zero components. Coordinate Descent is used to optimize single starlets. A proof-of-concept version of this algorithm was developed in python.

Although the algorithm produces super-resolved images in section 5, it is currently not known if it actually converges to the true optimum. Finding all relevant non-zero starlets is left to a greedy heuristic. There may be conditions under which it never includes relevant components. Further convergence analysis was outside the scope of the project. Also note that for this implementation, every time likely non-zero components are searched, it simply uses the non-uniform FFT for the dirty image approximation. This was done for simplicity's sake and can be improved.

## 4.1 The Starlet Transform

First we introduce the starlet transform, and then describe how we use it to estimate likely non-zero starlet components.

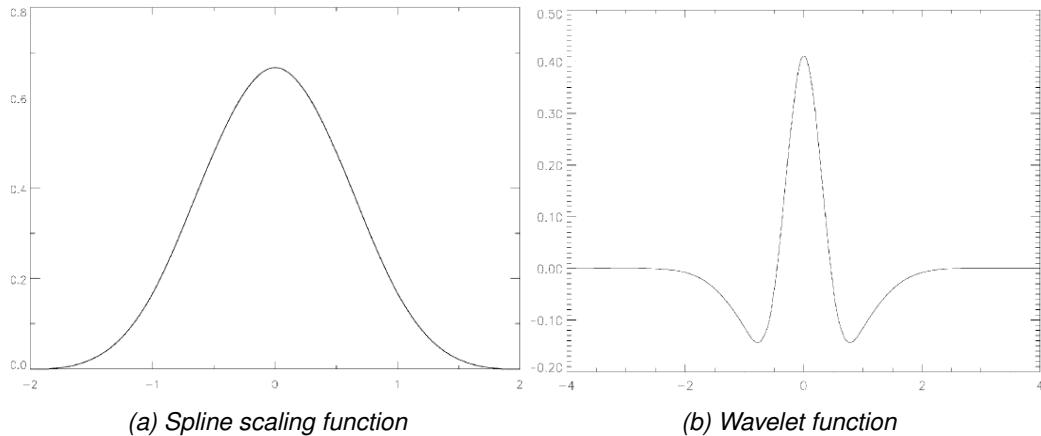


Figure 4: Scaling and wavelet function of the starlet wavelet. Source: [22]

The starlet transform is an Isotropic Undecimated Wavelet Transform (IUWT) with the starlet wavelet shown in figure 4. It defines two operations: The transformation from starlet into image space which is called synthesis, and the inverse which is called decomposition. The starlet space represents an image in multiple layers at different scales, where lower layers contain smaller objects and upper layers the extended emissions. The

lowest starlet layer represents the stars, while the upper layer represents the widest hydrogen clouds in the image.

$$\begin{aligned} c_0 &= \mathbf{x} \star B_0 \\ \mathbf{w}_0 &= \mathbf{x} - c_0 \end{aligned} \tag{4.2}$$

Let us look at the lowest starlet layer first, how it is calculated from the image, and how we use it for image reconstruction. The starlet layer  $w_0$  gets calculated in two steps, shown in (4.2). First, we take the scaling function for the lowest layer  $B_0$ , and convolve the image with it, resulting in the blurred image  $c_0$ . The blurring removes the smaller structures of the image. In the last step, we subtract the blurred image  $c_0$  from  $x$  and arrive at the starlet layer 0,  $w_0$ . It contains the smallest structure of the image convolved with the starlet wavelet.

This is where the starlet transform ends. Note that we do not get the actual starlet components  $\alpha_0$  for the layer. The transformation just decomposes the structures of the image into layers. The actual starlet components  $\alpha_0$  would be the result of a deconvolution.



(a) Dirty Image with two point sources.  
 (b) Layer  $w_0$  of the starlet decomposition.

*Figure 5: Dirty map and  $w_0$  starlet layer after shrinkage.  $w_0$  can be interpreted as a probability distribution for point source locations.*

It turns out we can estimate the starlet components  $\alpha_0$  by simply shrinking the starlet layer  $w_0$ . Figure 5 shows an example of the process. Coordinate Descent will reconstruct the image with starlet components  $\alpha_0$ , but it does not need to try all components. We use the starlet transform as a heuristic and can estimate likely non-zero  $\alpha_0$ .

The same process is repeated for all  $J$  layers. We calculate the starlet decomposition shown in equation (4.3). We continually blur the image with ever wider scaling functions  $B_i$ . We subtract the more heavily blurred image  $c_{J-1}$  from  $c_{J-2}$  to arrive at the starlet layer, which contains the structures of the current size  $w_{J-1}$ . The scaling function  $B_i$  widens exponentially, and the top level starlet spans an area of  $5 * 2^{J-1}$  pixels. Note that the last starlet layer is the most blurred version of the image  $c_J$ . It contains all structures which were wider than the widest starlet.

$$\begin{aligned} c_0 &= \mathbf{x} \star B_0 & c_1 &= c_0 \star B_1 & \dots & c_{J-1} &= c_{J-2} \star B_{J-1} & \mathbf{c}_J &= c_{J-1} \star B_J \\ \mathbf{w}_0 &= \mathbf{x} - c_0 & \mathbf{w}_1 &= c_0 - c_1 & \dots & \mathbf{w}_{J-1} &= c_{J-2} - c_{J-1} \end{aligned} \tag{4.3}$$

We shrink all the starlet layers  $w_i$  and  $c_J$ , and arrive at which components  $\alpha_i$  are likely non-zero. Coordinate Descent converges on the optimal  $\alpha_i$  for all layers. At this point, we can reconstruct the image by reversing

the process. We convolve  $\alpha_i$  with the starlet wavelet at scale i, which gives us  $w_i$ . Then, we sum them up to the final image (4.4).

$$x = w_0 + w_1 + \dots + w_{J-1} + c_J \quad (4.4)$$

This is how we leverage the starlet transform for our Coordinate Descent algorithm. Note that we can use the starlet transform as a heuristic without necessarily using the starlet components  $\alpha_i$  for reconstructions. The starlet transform gives us likely locations of Gaussian-shaped emissions at different scales. We can for example use a Gaussian mixture model, and just use the starlets for likely non-zero locations.

In this work, we do not use the wavelet shown in image 4b. Instead, we modified the starlet to an extend where it arguably is not a wavelet any more. Still, the starlet transform with the correct wavelet as a heuristic, but reconstructed the image with a modified starlet to satisfy the non-negativity constraint.

#### 4.1.1 Non-negative Starlet Modification

Reconstruction algorithms for Radio Astronomy typically constrain the image to be non-negative[10]. We can constrain the starlet components  $\alpha_i$  to be non-negative, but the starlet wavelet has negative sections. Which leads to negative areas in the layers  $w_0$ , and negative pixel values in the reconstructed image  $x$ .

The simple fix is to either set all negative pixels of  $x$  to zero, or to set negative values of  $w_i$  to zero. Both lead to slightly different artefacts in the  $x$ , and both do not lead to significantly better results.

For reconstruction, we modify the starlet wavelet and set all negative sections to zero. At this point, the reconstruction starlet is arguably not a wavelet any more, since its integral is larger than zero. But now the Coordinate Descent algorithm can reconstruct a non-negative image which has led to a more sparse representation of images in the next section.

## 4.2 Proof of concept Implementation

In this section, we put together the starlet transform and the Coordinate Descent to form our algorithm. We use two separate loops of Coordinate Descent minimization. First, let us look at the objective function (4.5) for a single starlet layer and discuss why we still need the non-uniform FFT for pre-calculation.

Our target is to use as few columns from  $F$  as possible. As we have seen in equation (4.3), each starlet is a series of convolutions. We can also do the convolution in Fourier space, and we only need to calculate the columns of  $F$  for non-zero starlets instead of non-zero pixels. This leads us to the following objective (4.5) for a single layer  $i$ .

$$\underset{\alpha_i}{\text{minimize}} \|V - S_i F \alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \quad (4.5)$$

The drawback is that we need to calculate all the convolutions for the starlets at layer  $i$  in the non-uniformly sampled Fourier space. We used the non-uniform FFT approximation to calculate all  $S_i$ . This introduces an approximation again. In the future, one can find the optimal  $S_i$  in the non-uniform Fourier space by solving an optimization problem similar to objective (3.2). The starlet is constant for a given observation, which means we can pre-calculate all  $S_i$  before the optimization.

#### 4.2.1 Iteration Scheme and Heuristics

Our algorithm is split in two separate Coordinate Descent loops. The first "adding" loop, iterates over the starlet components of a single layer. Here we use the starlet transform as a heuristic and add likely non-zero components. The second, "sparsifying" loop iterates over all non-zero components of all layers. Together they form one full iteration of our proof-of-concept algorithm.

```

1 def one_full_iteration(V_residual, alpha, J, lambda):
2     #adding loop
3     for j in reverse(range(0, J)):
4         a = alpha[j]
5         active_set = starlet_decomposition(non_uniform_fft(V_residual), lambda)
6         a = coordinate_descent(V_residual, active_set, lambda, 10)
7
8     #sparsifying loop
9     non_zero = sum(abs(alpha))
10    for i in range(0, 10):
11        for j in range(0, J):
12            a = alpha[j]
13            active_set = a[non_zero > 0]
14            a = coordinate_descent(V_residual, active_set, lambda, 1)

```

---

The adding loop minimizes the objective (4.5). We first use the starlet transform for estimating likely non-zero components. We then use Coordinate Descent and iterate over the likely non-zero components. In theory, the Coordinate Descent iterations in the adding loop are not necessary. The sparsifying loop will converge to the same solution. In practice, the Coordinate Descent iterations help to reduce the non-zero components we add. When we look again at image 5b, which shows the likely location of point sources, we see that the individual locations are not independent. Coordinate Descent iterations here help to reduce the number of non-zero starlets that get minimized in the sparsifying loop.

The adding loop minimized each layer separately. The task for the sparsifying loop is to minimize  $\alpha_i$  for all layers. In this implementation, we look at all  $\alpha_i$  for each non-zero location in  $\alpha$ . If the adding loop set a point source at a certain pixel location, the sparsifying loop checks if a wider starlet at the same location minimizes the objective.

The starlet transform allows for a myriad of heuristics with Coordinate Descent and is not limited to predict non-zero basis. In this project, we used the starlet transform to sort the active set in the adding loop. Coordinate Descent will first iterate over the most likely non-zero locations and continue its way downwards. In the example of image 5b we know it contains two point sources, and this heuristic lets Coordinate Descent check the correct pixel locations first before it looks at the area around the point source. At least for simulated data, this heuristic increases convergence speed.

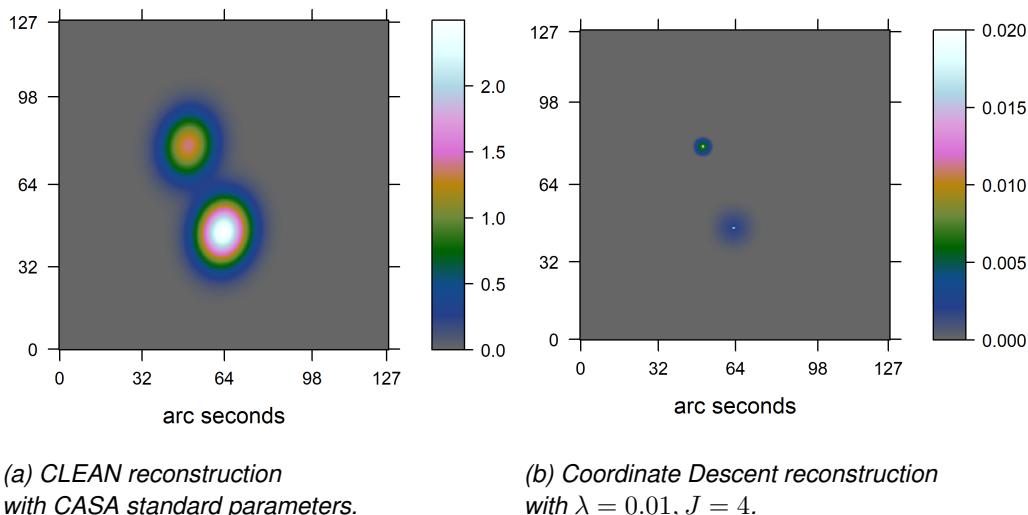
## 5 Test on simulated data

In this section, we test our Coordinate Descent, which uses the direct Fourier Transform, on simulated MeerKAT data. As mentioned in section 4, it is unknown if our approach will converge to the true optimum. Nevertheless, we compare our results with CASA's CLEAN implementation, and demonstrate super-resolution performance of Coordinate descent together with accurate total flux modelling.

The two simulated datasets contains idealized MeerKAT observations. Compared to the real world, the two simulated datasets contain few Visibilities and not representative of the real data volume. Also, more realistic simulations which contain pointing-, calibration-, and thermal noise are out of scope for this project. The simulations are used to isolate the two fundamental issues in radio interferometer image reconstruction: Non-uniform sampling and incomplete measurements.

### 5.1 Super-resolution of two point sources

The first simulated observation contains two non-zero pixels, i.e. point sources, with intensity of 2.5 and 1.4 Jansky/Beam. The image has a size of  $256^2$  at a resolution of 0.5 arc-seconds per pixel. The integral, the total Flux of the image, is 3.9 Jansky/beam.



*Figure 6: Image reconstruction of two simulated point sources.*

The figure 6 shows the CLEAN and the Coordinate Descent reconstruction. CLEAN reconstructs the image 6a at the accuracy limit of the instrument. It essentially reconstructs a blurred version of the observed image, where the blurring represents the accuracy of the instrument. With compressed sensing, we aim to reconstruct the de-blurred image, increasing the effective accuracy of the instrument.

Coordinate Descent in image 6b shows a super-resolved reconstruction of the two point sources. It reconstructs two narrow peaks surrounded by a low-intensity Gaussian emission. Also, Coordinate Descent manages to capture the total flux more accurately than CLEAN. The total flux of image 6b results in 3.92 Jansky/beam. CLEAN on the other hand overshoots the correct total flux by a factor of 1400. The total flux. It gets obvious when we compare the intensity profile of CLEAN, Coordinate Descent and the ground truth in figure 7.

CLEAN essentially places a Gaussian function with correct peak intensity at the point source location, but it does not respect the total flux of the image. However, note that Coordinate Descent in figure 7 seems to have



Figure 7: Intensity profile of the two point sources.

both point sources shifted by approximately a pixel. It looks suspiciously like an off-by one error. Sadly in the time frame of this project, no error was found or an explanation for this behaviour.

## 5.2 Super resolution of mixed sources

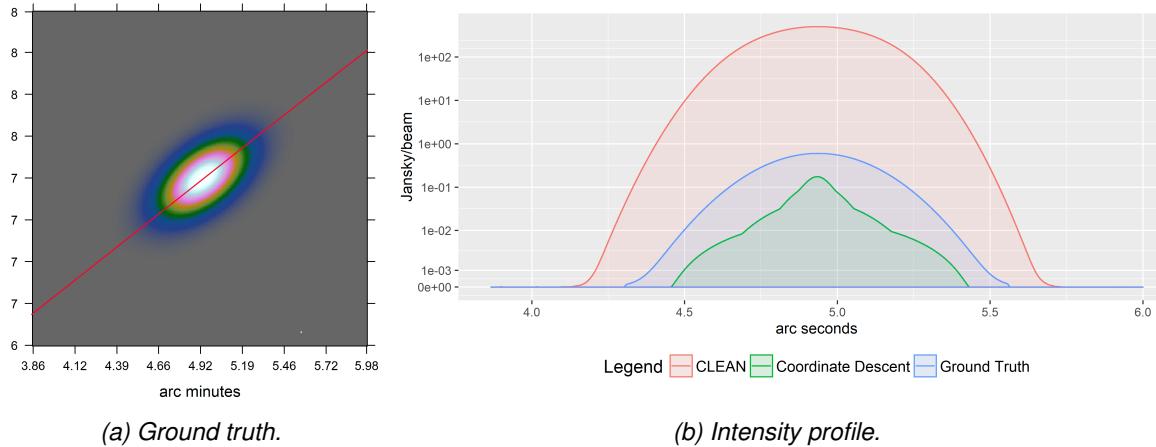
This dataset contains a mixture of three Gaussian emissions and sixteen point sources of varying intensities. At the center of the image, it has three point sources underlying a weak extended emission. The image center and one Gaussian emissions are analysed in detail. Coordinate Descent was run for two full iterations, using  $\lambda = 0.01$  and  $J = 7$  starlet layers. The CLEAN and Coordinate Descent reconstructions are shown in figure 8. Our algorithm was able to locate the point sources below the accuracy limit of MeerKAT. However, some point sources were reconstructed with artefacts. We analyse two regions of the image in more detail.



Figure 8: Reconstruction on mixed sources

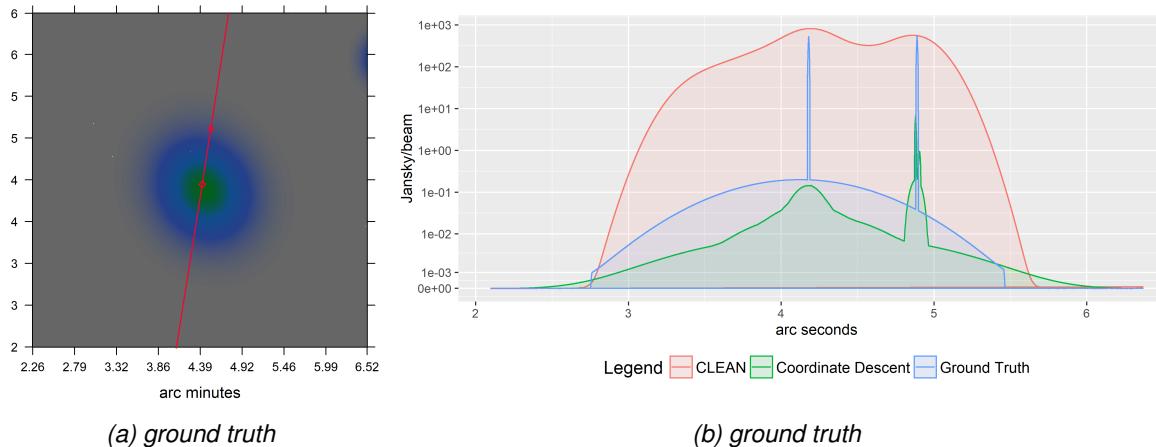
Let us look at the intensity profile of first region in figure 9. The Ground truth in image 9a shows a Gaussian extended emission with an oval shape. The figure 9b compares the intensity profile of Coordinate Descent with CLEAN and the ground truth. The intensity profile on the logarithmic scale emphasises the artefacts of the starlet reconstructions. Coordinate Descent used three starlets at different scales to approximate the extended emission. It reconstructed the flux more accurately, but did not capture the shape of the extended

emission. Instead, it approximated the oval shape with a uniform Gaussian-like emission. CLEAN on the other hand, did not again over-estimated the flux, but captured the oval shape more accurately.



*Figure 9: Intensity profile of region 1.*

The second region in figure 10 is more complex. It contains an extended Gaussian emission with several point sources. The intensity profile is cut through the center and two point sources shown in the ground truth image 10a. CLEAN again accurately reconstructs the peak intensity of the point sources, but over-estimates the total flux of the region. Coordinate Descent separates the point source from the extended emission. Sadly, the first point source seems again shifted by a pixel, while the second point source was reconstructed with three peaks at varying intensities.



*Figure 10: Intensity profile of region 2.*

Compared to CLEAN, Coordinate Descent with the modified starlet regularization reconstructed the image with a more accurate total flux, and located the point sources below the accuracy of the instrument. The quality of the point source reconstruction varies within the image 8b. The figure 11 shows four different point sources of in detail. From left to right, the reconstruction quality of each example decreases.

The left image shows shows a reconstruction where the flux is concentrated into a single pixel, and has low intensity emission around. Our modified starlet reconstruction should produce point sources similar to it. The second image from the left looks similar at first glance, but contains a lower intensity "trail". The third and fourth image even contain a fake point source in close proximity.

Our modified starlet has likely a role in these artefacts. In section 4.1.1, we described how our modified starlet does not have negative sections. Intuitively, the original starlet penalizes two point sources in close proximity,



Figure 11: Different point source reconstructions of Coordinate Descent.

and forces the emission to be represented with a wider starlet. Since we removed the negative sections, our algorithm can use as many point sources next to each other without any penalty. This would explain the "trail" of single pixel emissions in the second, third and fourth images.

The original starlet wavelet forces the second, third and fourth image of figure 11 to be similar to the first, but at the cost of sparsity. Our modified starlet can represent the image 8b with 244 non-zero bases, while the original starlet leads to over 2000. It has to correct for the negative image regions it produces.

Our modified starlet leads to a more sparse solution, but may introduce artefacts in reconstructing point sources. In our proof-of-concept-implementation, we used the same  $\lambda$  regularization for each layer. Girard et al.[5] used a different  $\lambda$  for every layer. It may help reduce the artefacts of our modified starlet, while keeping the sparse representation it provides.

## 6 Runtime cost comparison on a real-world MeerKAT reconstruction

Current Compressed Sensing reconstructions produce images at a higher quality than CLEAN. However, CLEAN is significantly cheaper to compute. MeerKAT's large scale reconstruction problems, CLEAN is still the go-to algorithm. In this project, we developed for a new architecture, which uses the relevant columns of the Fourier Transform Matrix directly. We developed a Coordinate Descent algorithm with this architecture and demonstrated in section 5 super-resolution performance on simulated data. The question, if we can lower the runtime costs with our new architecture, is still open.

In this section, we compare the costs of Coordinate Descent with WSCLEAN, which is the reconstruction algorithm of choice for MeerKAT reconstructions. We create cost functions for each algorithm, which estimate the number of operations depending on the input size. WSCLEAN was executed on a real-world MeerKAT dataset shown in image 12. Our proof-of-concept implementation was not able to handle the large amount of data. Instead, we extrapolate the best-case costs of our approach and compare them to WSCLEAN on the MeerKAT dataset.

### 6.1 Cost function of an idealized Coordinate Descent

The runtime cost of Coordinate Descent depends on the number of Visibilities  $M$  and the number of non-zero starlets  $S$ . The number and location of the  $S$  non-zero starlets are generally not known. However, we created a heuristic which finds likely non-zero starlet components. In a realistic setting, the heuristic will have found more than  $S$  likely non-zero starlets. For the idealized version of Coordinate Descent, we assume an oracle performance heuristic: It finds the location and number of the  $S$  non-zero starlet components in constant time. Coordinate Descent therefore has to calculate the value of  $S$  components. In total, the idealized Coordinate Descent algorithm uses four steps: creating  $J$  starlet levels with the non-uniform FFT, creating the columns of  $F^{-1}$ , calculating the minima for each single component, and calculating the starlet layers:

$$\begin{aligned}
 & J \text{ non-uniform FFTs for the starlet regularization : } J * (M + 2N * \text{ld}(2N)) \\
 & \quad \text{creating } S \text{ columns of } F^{-1} : S * 7M \\
 & \quad \text{locating } S \text{ minima of } S \text{ parabolas : } S * 4M \\
 & \quad \text{calculating } J \text{ Starlet layers : } J * 2M
 \end{aligned}$$

We assume we have enough memory to cache the columns of  $F^{-1}$  and only need to calculate them once. Keep in mind that each column of  $F^{-1}$  has the same length as the Visibilities, essentially multiplying the input data. The last parameter for Coordinate Descent is the number of iterations to converge,  $I_{CD}$ . Estimating this number is difficult as Coordinate Descent does not have strict guarantees (as discussed in section 4). Instead, we assume it converges after a fixed number of iterations. Therefore we arrive at the cost function of (6.1).

$$\begin{aligned}
 CD(I_{CD}, M, S, J) = & I_{CD} * [S * 4M + J * 2M] \\
 & + S * 7M \\
 & + J * (M + 2N * \text{ld}(2N))
 \end{aligned} \tag{6.1}$$

Note that the runtime of Coordinate Descent is independent of the number of pixels. The only image related parameter in (6.1) is  $J$ , the number of starlet layers. The largest starlet layer represents the largest possible structure in the image, which is given by the instrument and the image resolution. The runtime only depends indirectly on the image resolution, not the total number of pixels. For simplicity, we assume the image cannot

have structures larger than half the image size. For our MeerKAT example, this is more than enough to represent the largest structures.

Also note the term iterating over the  $S$  non-zero starlets,  $I_{CD} * [S * 4M + \dots]$ . As it turns out, this is the Achilles heel of the algorithm. MeerKAT observations contain a very large amount of Visibilities  $M$ .

## 6.2 Cost function of WSCLEAN

The WSCLEAN algorithm uses the Major Cycle architecture. It uses the non-uniform FFT with  $w$ -stacking. The runtime costs of a single Major Cycle depends on the non-uniform FFT with  $w$ -stacking and the number of CLEAN deconvolutions.  $N$  denotes the number of pixels.

$$\begin{aligned} \text{non-uniform FFT} &: M + 2N * ld(2N) \\ \text{non-uniform FFT with } w\text{-stacking} &: M + W * (2N * ld(2N) + 2N) + N * ld(N) \\ I_{CLEAN} \text{ deconvolutions} &: I_{WSCLEAN} * 2N \end{aligned}$$

The overall cost function shown in (6.2) can also be split into two parts. In each Major Cycle, the forward and backwards non-uniform FFTs gets calculated, and CLEAN deconvolves the image for a maximum number of  $I_{CLEAN}$  deconvolutions.

$$\begin{aligned} WSCLEAN(I_{Major}, I_{CLEAN}, M, N, W) &= I_{Major} * 2 * [M + W * (2N * ld(2N) + 2N) + N \log N] \\ &\quad + [I_{CLEAN} * 2N] \end{aligned} \tag{6.2}$$

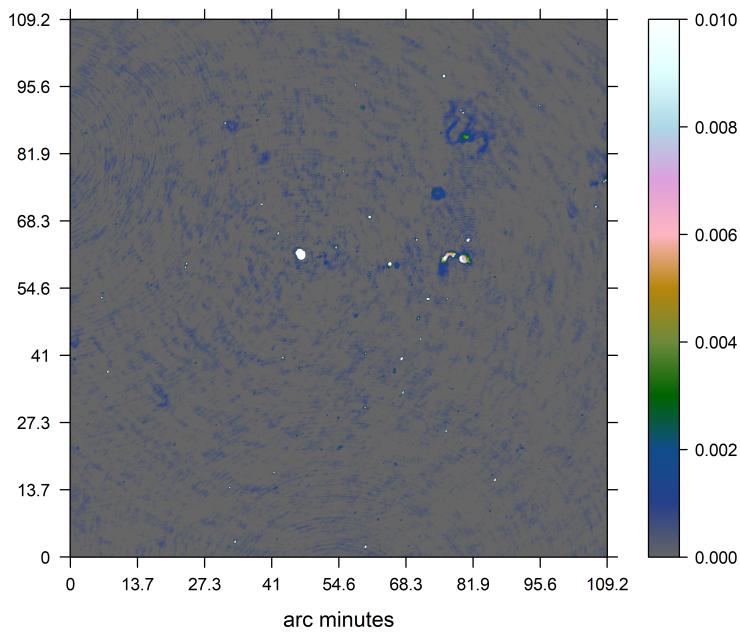
Notice that the number of CLEAN deconvolutions  $I_{CLEAN}$  depends on the image content, similar the number of non-zero starlets  $S$  for Coordinate Descent. Here however, it multiplies with the number of pixels instead of the number of Visibilities. In a sense, the major cycle tries to reduce the runtime complexity of handling the image content by calculating the non-uniform FFT. If the difference is large enough  $N \ll M$ , then the Major Cycle will end up with a smaller runtime costs.

## 6.3 Comparison on a MeerKAT reconstruction problem

Our real-world MeerKAT observation has been calibrated and averaged in frequency and time to reduce storage space. The resulting dataset contains 540 channels with 4 million Visibilities each. Due to hardware limitations, WSCLEAN was calculated on 75 channels. The reconstructed image is shown in 12, and the resulting parameters for our cost function are:

- Major Cycles:  $I_{Major} = 6$
- Number of CLEAN iterations:  $I_{CLEAN} = 35'000$
- Visibilities:  $M = 3.05e^8$
- Pixels:  $N = 2048^2$
- $w$ -stacks:  $W = 32$

For Coordinate Descent's costs, we need an estimate for  $J$ ,  $S$  and  $I_{CD}$ . We set  $J = 8$ , which lets the largest structure span over half the image, enough to capture any large scale structures. For  $S$  and  $I_{CD}$ , we simply



*Figure 12: WSCLEAN Reconstruction of the MeerKAT observation.*

use the values from our simulated reconstruction 8b and set  $S = 250$  and  $I_{CD} = 10$ . This is an underestimation of the true values. The image 12 shows complex-shaped extended emissions, which likely needs a larger number of starlets for representation than the Gaussian emissions from our simulation.

When we put all values into our cost functions (6.1) and (6.2), Coordinate Descent with the direct Fourier transform arrives at 9.7 times the costs of WSCLEAN in the best case scenario. Our approach has not reduced the runtime costs compared to WSCLEAN. But what about Major Cycle Compressed Sensing algorithms? We have not developed a cost function for other Major Cycle approaches, but we can get a very rough estimate by changing  $I_{Major}$  of the WSCLEAN cost function. Pratley et al.[12] reported 10 Major Cycles for a Compressed Sensing reconstruction on simulated data. If we plug in 10 Major Cycles for WSCLEAN, Coordinate Descent ends up taking roughly 8 times the cost of the Major Cycle algorithm. Mind you these costs are only valid when we can keep all necessary columns of  $F$  in memory, eating up 1.1 terabytes<sup>4</sup> in our case. If not, the columns have to be re-calculated on the fly, increasing the runtime costs by magnitudes.

The issue with Coordinate Descent's runtime complexity lies in the term  $I_{CD} * [S * 4M + \dots]$  of (6.1), which scales with the "content" of the image  $S$ , multiplied with the Visibilities. Coordinate Descent cannot afford many iterations nor many non-zero components, because both of these numbers get multiplied together with  $M$ , the largest number in the problem. With the Major Cycle architecture, WSCLEAN is able to get around this limitation, and scales any content dependent factors on  $N$  instead of  $M$ .

Indeed, the runtime of our Coordinate Descent algorithm could be improved by using the Major Cycle architecture, essentially replacing  $M$  with  $N$  and we arrive at the term  $I_{CD} * [S * 4N + \dots]$ . In this case Coordinate Descent can afford more iterations and more non-zero components for the same runtime costs. Furthermore  $N$  lies on a uniformly sampled grid. We may be able to use the FFT instead of caching columns of  $F$ , and reduce the memory requirement at the same time.

---

<sup>4</sup>If we assume 64 bit floating point accuracy for the real and complex values of the Visibilities.

## 6.4 Approximations as key for going large scale

We set out to reduce the runtime costs of compressed Sensing reconstructions by replacing the non-uniform FFT approximation. Ironically we ended up with an algorithm which not only has higher runtime costs, the costs may even be reduced by moving back to the Major Cycle architecture.

The question arises, whether the direct Fourier Transform has a future for MeerKAT reconstructions or not. From the current results we can summarize that the Major Cycle architecture with the non-uniform FFT is a good approximation of the Fourier Transform with no clear alternative. The emphasis is on the approximation. We do not need an exact transform. The Visibilities are inherently noisy, and even with the optimal reconstruction there will be residual noise in the image above the level of floating point accuracies. An optimal Fourier Transform for MeerKAT takes the inherent inaccuracy into account.

The direct Fourier Transform, as it is implemented in our algorithm, wastes resources with a precise transformation. In the context of MeerKAT, it has to re-introduce approximations to be competitive. Our Coordinate Descent approach leveraged the fact that not all columns of  $F$  are necessary for reconstruction. The next step is to see if the number of rows can be reduced too.

A MeerKAT reconstruction has magnitudes fewer pixels than Visibilities. Not all samples contribute the same amount of information to a reconstruction. If we can remove the redundant information before reconstruction, we can reduce the overall costs and increase the ratio of samples per pixel. Our Coordinate Descent algorithm scales independently of image size, the fewer samples we have per pixel, the more cost effective our algorithm becomes compared to CLEAN.

As it is, our proof-of-concept algorithm with the direct Fourier Transform is too expensive when compared to CLEAN. It wastes resources on a too precise transformation. By re-introducing approximations, we can further decrease the runtime costs of the direct Fourier Transform. Whether we can find an approximation, which is at least as accurate as the non-uniform FFT and has lower runtime costs remains to be investigated in the future.

## 7 Compressed Sensing reconstructions in the MeerKAT era

Current Compressed Sensing reconstructions share the non-uniform FFT approximation with CLEAN. Both continually cycle between Visibility and image space. Compared to CLEAN, Compressed Sensing algorithms need more cycles to converge, which is one of the reasons why they have higher runtime costs. In this project, we postulated that one may reduce the runtime cost of Compressed Sensing reconstructions by replacing the non-uniform FFT approximation.

We discussed three alternatives: Optimal Projection on a uniform Grid, using Spherical Harmonics and the direct Fourier Transform.

With the latter approach, we created a proof-of-concept image reconstruction using Coordinate Descent. We leveraged the starlet transform and created an algorithm which scales with the number of non-zero basis functions. We demonstrated super-resolution performance on simulated MeerKAT data and extrapolated the runtime costs on a real-world observation. Compared to CLEAN and the non-uniform FFT, our algorithm leads to higher runtime costs. As it is implemented in this project, the direct Fourier Transform is not a viable alternative on MeerKAT data.

Optimal Projection is another alternative to the non-uniform FFT and the direct Fourier Transform. It is way of reducing the problem to a more manageable size, since the total grid size is usually a fraction of the total measurements. In this architecture, the first step is to search for the optimal projection of Visibilities on a uniform grid. We then reconstruct the image with uniformly sampled Visibilities and do not need the original measurements. However, self-calibration requires a transformation from uniform sampled grid back to the original measurements. In this task, we cannot ignore the original Visibilities in later processing. Unless one finds a way to project the calibration problem on a uniform grid too, self-calibration destroys the main advantage of the approach.

The other alternative is using Spherical Harmonics instead of the Fourier Transform, which is a new way to represent the Radio Interferometric Measurement Equation. We analysed published research in this area[14, 16] and see potential for a new reconstruction architecture, which may reduce the overall runtime costs. However, the current formalism[18] is based on the plane wave. How a real-world imaging algorithm with spherical harmonics deals with calibration and ionosphere distortion may be fundamentally different to current CLEAN and Compressed Sensing reconstructions. At this point, it is unclear if or how many imaging problems have to be re-investigated by moving to Spherical Harmonics.

In this project we have not found a clear alternative to the non-uniform FFT. It leads to an iterative reconstruction algorithm with overall lower runtime costs. However, the iterative nature becomes troublesome for large scale distribution. New research has found ways to distribute parts of the algorithm[12, 23], but how to distribute the reconstruction on a large scale is still an open problem.

The direct Fourier Transform does not need iterative approximation algorithms and lends itself to distributed computing.

One advantage of the direct Fourier Transform is it simplifies distributed computing. The drawback of the direct Fourier Transform is that on MeerKAT data, it has an impractically large memory requirement or results in higher runtime costs. Our Coordinate Descent approach significantly reduces the memory requirement and runtime costs of the direct Fourier Transform. It is a step towards making the direct Fourier Transform more competitive, but our improvement alone is not enough. The memory requirement and runtime costs may be further reduced by leveraging the inherent redundancy in MeerKAT measurements. If this leads to a comparable runtime cost to the non-uniform FFT, the direct Fourier Transform becomes interesting for distributed computing.

## References

- [1] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.
- [2] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.
- [3] JW Rich, WJG De Blok, TJ Cornwell, Elias Brinks, Fabian Walter, Ioannis Bagetakos, and RC Kennicutt Jr. Multi-scale clean: A comparison of its performance against classical clean on galaxies using things. *The Astronomical Journal*, 136(6):2897, 2008.
- [4] Urvashi Rau and Tim J Cornwell. A multi-scale multi-frequency deconvolution algorithm for synthesis imaging in radio interferometry. *Astronomy & Astrophysics*, 532:A71, 2011.
- [5] Julien N Girard, Hugh Garsden, Jean Luc Starck, Stéphane Corbel, Arnaud Woiselle, Cyril Tasse, John P McKean, and Jérôme Bobin. Sparse representations and convex optimization as tools for lofar radio interferometric imaging. *Journal of Instrumentation*, 10(08):C08013, 2015.
- [6] Arwa Dabbech, Alexandru Onose, Abdullah Abdulaziz, Richard A Perley, Oleg M Smirnov, and Yves Wiaux. Cygnus a super-resolved via convex optimization from vla data. *Monthly Notices of the Royal Astronomical Society*, 476(3):2853–2866, 2018.
- [7] Stefan Kunis, J Keiner, and D Potts. Nonequispaced fft. Available: [www-user.tu-chemnitz.de/skunis/paper/lecturefft.pdf](http://www-user.tu-chemnitz.de/skunis/paper/lecturefft.pdf).
- [8] Luke Pratley, Jason D McEwen, Mayeul d’Avezac, Rafael E Carrillo, Alexandru Onose, and Yves Wiaux. Robust sparse image reconstruction of radio interferometric observations with purify. *Monthly Notices of the Royal Astronomical Society*, 473(1):1038–1058, 2017.
- [9] Tim J Cornwell, Kumar Golap, and Sanjay Bhatnagar. The noncoplanar baselines effect in radio interferometry: The w-projection algorithm. *IEEE Journal of Selected Topics in Signal Processing*, 2(5):647–657, 2008.
- [10] Jason D McEwen and Yves Wiaux. Compressed sensing for wide-field radio interferometric imaging. *Monthly Notices of the Royal Astronomical Society*, 413(2):1318–1332, 2011.
- [11] HT Intema, S Van der Tol, WD Cotton, AS Cohen, IM Van Bommel, and HJA Röttgering. Ionospheric calibration of low frequency radio interferometric observations using the peeling scheme-i. method description and first results. *Astronomy & Astrophysics*, 501(3):1185–1205, 2009.
- [12] Luke Pratley, Melanie Johnston-Hollitt, and Jason D McEwen. A fast and exact *w*-stacking and *w*-projection hybrid algorithm for wide-field interferometric imaging. [arXiv preprint arXiv:1807.09239](https://arxiv.org/abs/1807.09239), 2018.
- [13] Arwa Dabbech, Laura Wolz, Luke Pratley, Jason D McEwen, and Yves Wiaux. The *w*-effect in interferometric imaging: from a fast sparse measurement operator to superresolution. *Monthly Notices of the Royal Astronomical Society*, 471(4):4300–4313, 2017.
- [14] TD Carozzi. Imaging on a sphere with interferometers: the spherical wave harmonic transform. *Monthly Notices of the Royal Astronomical Society: Letters*, 451(1):L6–L10, 2015.
- [15] Nathanaël Schaeffer. Efficient spherical harmonic transforms aimed at pseudospectral numerical simulations. *Geochemistry, Geophysics, Geosystems*, 14(3):751–758, 2013.

- [16] JD McEwen and AMM Scaife. Simulating full-sky interferometric observations. *Monthly Notices of the Royal Astronomical Society*, 389(3):1163–1178, 2008.
- [17] Anthony Richard Thompson, James M Moran, George Warner Swenson, et al. *Interferometry and synthesis in radio astronomy*. Springer, 1986.
- [18] Oleg M Smirnov. Revisiting the radio interferometer measurement equation-i. a full-sky jones formalism. *Astronomy & Astrophysics*, 527:A106, 2011.
- [19] Stephen J Hardy. Direct deconvolution of radio synthesis images using l1 minimisation. *Astronomy & Astrophysics*, 557:A134, 2013.
- [20] Zhi-Quan Luo and Paul Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.
- [21] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.
- [22] Jean-Luc Starck, Fionn Murtagh, and Mario Bertero. Starlet transform in astronomical data processing. *Handbook of Mathematical Methods in Imaging*, pages 2053–2098, 2015.
- [23] AR Offringa, Benjamin McKinley, Natasha Hurley-Walker, FH Briggs, RB Wayth, DL Kaplan, ME Bell, Lu Feng, AR Neben, JD Hughes, et al. Wsclean: an implementation of a fast, generic wide-field imager for radio astronomy. *Monthly Notices of the Royal Astronomical Society*, 444(1):606–619, 2014.

## List of Figures

1	The Major Cycle Framework . . . . .	3
2	Celestial sphere distortion on simulated data. Source: [9] . . . . .	4
3	The Major Cycle Framework . . . . .	5
4	Scaling and wavelet function of the starlet wavelet. Source: [22] . . . . .	11
5	Dirty map and $w_0$ starlet layer after shrinkage. $w_0$ can be interpreted as a probability distribution for point source locations. . . . .	12
6	Image reconstruction of two simulated point sources. . . . .	15
7	Intensity profile of the two point sources. . . . .	16
8	Reconstruction on mixed sources . . . . .	16
9	Intensity profile of region 1. . . . .	17
10	Intensity profile of region 2. . . . .	17
11	Different point source reconstructions of Coordinate Descent. . . . .	18
12	WSCLEAN Reconstruction of the MeerKAT observation. . . . .	21

## List of Tables

## 8 Ehrlichkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende schriftliche Arbeit selbstständig und nur unter Zuhilfenahme der in den Verzeichnissen oder in den Anmerkungen genannten Quellen angefertigt habe. Ich versichere zudem, diese Arbeit nicht bereits anderweitig als Leistungsnachweis verwendet zu haben. Eine Überprüfung der Arbeit auf Plagiate unter Einsatz entsprechender Software darf vorgenommen werden.

Windisch, January 31, 2019

Jonas Schwammburger