

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

Jonas Schwammberger

January 23, 2019

Abstract

MeerKAT new Radio Interferometer, poses a large scale image reconstruction problem.

Compressed sensing reconstructions for Radio astronomy exist, but are slower than CLEAN.

Compressed Sensing and CLEAN use the same architecture, both use Major Cycle. In this work, we explore if Compressed Sensing reconstructions can be made faster with in a different architecture

Several possibilites are discussed and a new proof-of-concept algorithm is developed, which does not require a Major Cycle. We estimate the minimum runtime complexity of the algorithm and compare it to CLEAN.

However, the algorithm is not suitable in the context of meerkat reconstructions. It requires too much memory and

we come to the conclusion that in the context of MeerKAT reconstructions, the Major Cycle architecture is easier to compute.

No obvious contender to replace the Major Cycle.

Contents

1 Compressed Sensing Image Reconstruction for MeerKAT	1
1.1 The basic Measurement Equation of a radio interferometry	1
1.2 The Major Cycle Architecture	2
2 Challenges for imaging MeerKAT data	4
2.1 Wide Field of View Imaging and the third Fourier Component	4
2.1.1 State of the art: w -stacking algorithm and WSCLEAN	5
2.2 Self-Calibration	6
3 Eliminating the Major Cycle	7
3.1 Separating the Major Cycle into two Optimization Problems	7
3.2 Reconstruction on the Sphere	8
3.3 Direct Fourier Transform	8
4 Compressed Sensing with direct Fourier Transform and Coordinate Descent	10
4.1 The Starlet Transform	11
4.1.1 Handling the positivity constraint	13
4.2 Coordinate Descent Implementation	13
4.3 Iteration scheme	13
4.3.1 Non-uniform FFT approximations	14
5 Test on simulated data	15
5.1 Super-resolution of two point sources	15
5.2 Super resolution of mixed sources	16
6 Runtime cost comparison on a real-world MeerKAT reconstruction	18
6.1 Cost function of an idealized Coordinate Descent	18
6.2 Cost function of WSCLEAN	19
6.3 Comparison on a MeerKAT reconstruction problem	19
6.4 Approximations as key for going large scale	21
7 Compressed Sensing reconstructions in the MeerKAT era	22
8 Ehrlichkeitserklärung	26

1 Compressed Sensing Image Reconstruction for MeerKAT

An instrument in the real world measures noisy data. Measurements are corrupted by noise, interference sources or the measurement instrument itself. Image reconstruction problems appear when one tries to remove the corruption from the measurements and tries to find the observed image of the instrument. This leads to an ill-posed inverse problem: A small change in the measurements may create very different image reconstructions, and many possible images match the measurements. An image reconstruction algorithm therefore has to find the observed image from a potentially large set of possible images.

In the past, image reconstructions applied simple heuristics and approximated a likely image. How close the approximation was to the observed image was in general not known. The theory of compressed sensing[1][2] introduced a new theoretical framework under which image reconstructions can be analysed. This has lead rise to new compressed sensing reconstruction algorithms, which under the right conditions are guaranteed to reconstruct the observed image. Furthermore they have shown super-resolution performance in real-world environments, creating reconstructions above the accuracy limit of the instrument.

This work applies compressed sensing image reconstruction to the field of radio astronomy. The new MeerKAT radio interferometer poses a reconstruction problem on a new scale of data volume. The raw measurements easily take up several terabytes of disk space. The focus of this work is creating a scalable compressed sensing image reconstruction algorithm.

The current state of the art reconstruction algorithm for MeerKAT is based on CLEAN[3][4]. It is a reconstruction using a simple heuristic and was developed before the theory of compressed sensing was known. In recent years, compressed sensing reconstruction algorithms were developed for radio interferometers[5][6][7]. They beat CLEAN in terms of reconstruction quality, producing super-resolved reconstructions. However, CLEAN has the upper hand in runtime complexity. Therefore, an efficient implementation of CLEAN is still the go to reconstruction algorithm for MeerKAT data.

The efficient implementation use CLEAN in the Major Cycle Architecture, which was developed with CLEAN in mind. Current compressed sensing algorithms use a similar architecture. So far, little research has gone into different architectures for compressed sensing reconstructions in radio astronomy. This work explores different architectures for compressed sensing reconstructions with the hope of reducing the computational complexity.

A new proof-of-concept reconstruction algorithm was developed with a simplified architecture. It was tested on simulated MeerKAT data and the lower-bound asymptotic complexity was evaluated. The new algorithm scales independently of the image size, but scales worse with the number of input measurements compared to CLEAN. For the MeerKAT image reconstruction, the number of input measurements tends to be the largest of all numbers in the problem. Even though the algorithm could be improved further, it is unlikely to beat CLEAN reconstructions in terms of runtime complexity on MeerKAT data.

1.1 The basic Measurement Equation of a radio interferometry

Real world radio interferometers have complicated measurement equations. They become even more complicated for large interferometers like MeerKAT. These problems get addressed in section 2. This section looks at the basic measurement equation of a radio interferometer (1.1) and discusses the two fundamental challenges for radio interferometry image reconstruction.

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

An interferometer measures Fourier Components V (called Visibilities in Radio Astronomy) from the sky image I at position x and y . The term $e^{2\pi i(ux+vy)}$ represents the two dimensional Fourier Transform. The task is to reconstruct the observed image I from the measured Visibilities V . In theory this task is trivial: Since the inverse Fourier Transform exists, we can reconstruct the image I by calculating the inverse Fourier Transform of V . However, two properties of the Visibilities make this task challenging in practice:

1. Non-uniform sampling pattern in Visibility space
2. Incomplete Visibility coverage.

Property 1: We want to reconstruct an image with uniformly spaced pixels. The instrument defines the sampling pattern in Visibility space and does not correspond to the exact pixels of the reconstructed image. This property keeps us from using the Fast Fourier Transform. The naive inverse Fourier Transform can still be calculated, but it has a quadratic runtime and does not scale to the data volume of interferometers. Current reconstruction algorithms use the non-uniform Fast Fourier Transform. The non-uniform FFT approximates the non-uniform Fourier Transform.

Property 2: Interferometers sample only a limited set of Visibilities. It does not have all information for reconstruction. When the inverse Fourier Transform is applied on the Visibilities, the resulting image is corrupted by the incomplete Visibilities. It contains structures which were introduced by the interferometer and were not observed. With only knowing the incomplete set of Visibilities a reconstruction algorithm has to decide which image structures were truly measured, and which are due to the instrument. This forms an ill-posed inverse problem. There are many images that fit the measurements, and a small change in the Visibilities can lead to a very different reconstruction.

CLEAN represents the instrumental effect with a Point Spread Function (PSF). After the non-uniform FFT produced the 'dirty' image, CLEAN tries to reconstruct observed image with a deconvolving the 'dirty' image with the PSF. Note however that both CLEAN nor the non-uniform FFT are approximations. In real world reconstructions, these two approximations are used in the major cycle architecture to increase the reconstruction accuracy.

1.2 The Major Cycle Architecture

Major cycle was created with CLEAN in mind. Compressed sensing reconstructions use essentially the same architecture with minor modifications.

A CLEAN image reconstruction for radio interferometers consists of two different steps: A non-uniform FFT, which approximates the inverse Fourier Transform efficiently and an deconvolution algorithm, which approximates the instrumental effects on the image (typically based on CLEAN). These two approximations are an error source of the reconstruction. The major cycle architecture 1 therefore tries to iteratively minimize the errors of the non-uniform FFT and the deconvolution over several iterations.

The first major cycle uses the non-uniform inverse FFT to approximate the 'dirty' image from the Visibilities. The deconvolution algorithm decides parts which belong to the observed image and which are due to instrumental and other noise effects. It returns the noise part of the image, which gets transformed back into residual Visibilities. The next major cycle iteration continues with the

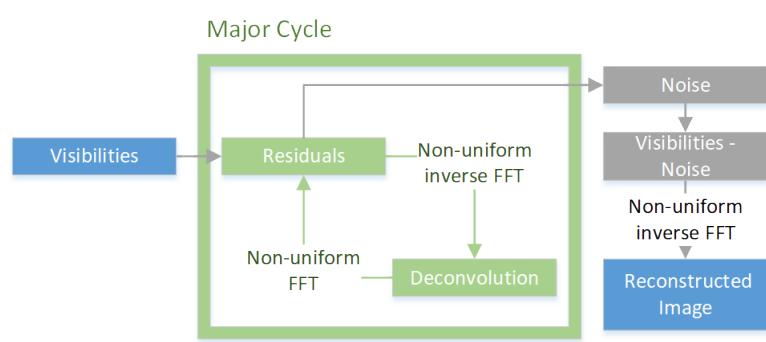


Figure 1: The Major Cycle Framework

residual Visibilities. The residual Visibilities get minimized until they contain only the instrumental effects and noise.

After several major cycles the residual on a regularly spaced image which has a small error from non-uniform samples, and a small error from incomplete measurements.

Compressed sensing reconstructions essentially use the same architecture, although with minor alterations. In general, they keep a similar scheme of forward/backward non-uniform FFT, but do not use a deconvolution in image pace. Instead, they analyse constraints defined on the image (For example all pixels should be non-negative), and try to find the Visibilities that minimize the constraint violations.

For CLEAN reconstructions, the forward/backward non-uniform FFT are the most expensive operations, while the deconvolution is negligible. The compressed sensing algorithms tend to require more major cycle iterations to converge and the image constraint analysis tends to be more expensive than CLEAN deconvolutions. Current research into compressed sensing reconstructions is focussed on reducing the number of major cycles[6].

2 Challenges for imaging MeerKAT data

New interferometers like MeerKAT put additional challenges on the image reconstruction problem. First, the new instrument produce a new magnitude of data, forcing reconstruction algorithms to be highly scalable and distributable. Second, the more sensitive instruments with large field of view amplify effects which were negligible in older instruments, like non-coplanar baselines or the ionosphere.

Compressed sensing has to be able to work with these issues somehow. Solutions all in the context of the major cycle, a new architecture may need to handle effects differently. Hopefully in a manner that is at least as efficient.

In this work, the effects of non-coplanar baselines gets handled in more detail. The effect has a neat mathematical notation, it adds a third fourier component to the measurement equation, but breaks the two dimensional fourier relationship introduced in section 1.1.

Calibration used to be a task before image reconstruction. Calibration gets not explicitly handled here, but it is important to keep in mind. For older interferometers, the data was calibrated before image reconstruction. With the advent of self-calibration, the image reconstruction is also used to improve calibration. MeerKAT requires more calibration parameters, so an image reconstruction algorithm has to be able to calibrate, or it is not interesting.

Further issues that do not get handled here

- (Beam Pattern, A Projection)
- Full polarization
- Wide band imaging

There are several challenger for imaging meerkat data. One problem is the new amount of data.

terabytes of measurements. Large image size 32k squared are the obvious problems to solve. Distributing the problem is not part of this work.

In this work, it is focused on Wide field of view issue.

2.1 Wide Field of View Imaging and the third Fourier Component

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 -component. 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. For wide field of view however, the w -component cannot be ignored. The figure 2b shows the effect of ignoring w . The image gets continually more distorted away from the center. Emissions at the edges of the image get "torn" apart.

We find the reason for why it distorts the edges, if we look at the measurement equation (2.1) in detail. We see that the image $I(x, y)$ 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 -component). More precisely, the instrument looks at the inside wall of the celestial sphere. The two dimensional Fourier transform

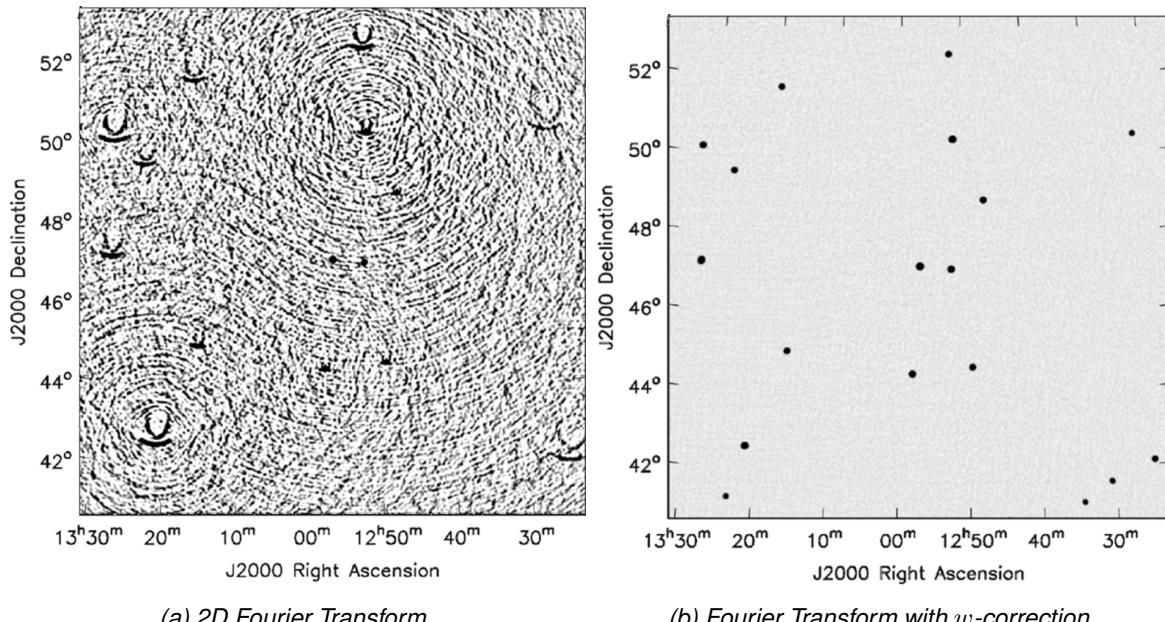


Figure 2: Celestial sphere distortion on simulated data. Source: [8]

approximates the sphere 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).

In the context of MeerKAT, the reconstruction algorithm has to handle the w -term in an efficient manner. 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, w -stacking is the state-of-the-art way of correcting for the w -component efficiently. Since this project explores different architecture, we may need to find another way of correcting for the w -term.

2.1.1 State of the art: w -stacking algorithm and WSCLEAN

w -stacking[9] is the current state-of-the-art approach for w -corrections. It is implemented in the WSCLEAN software package and is the current go-to algorithm for MeerKAT reconstructions. With w -stacking, we can again use the two dimensional non-uniform FFT for efficient transformations between image and Visibilities.

w -stacking, shown in figure 3 replaces the non-uniform FFT operation from figure 1. 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, summing over all stacked dirty like CLEAN can deconvolve the

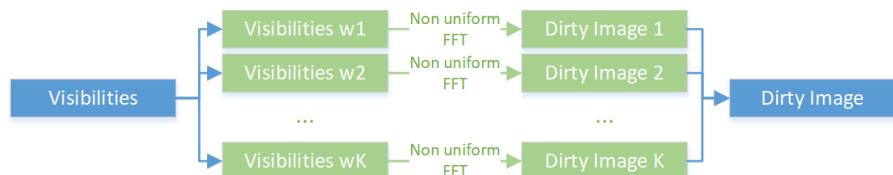


Figure 3: The Major Cycle Framework

uniform FFT. For each stack, the w -correction is performed for the center w -component. The last step is summing over all stacked dirty images, giving us the final dirty image. At this point a deconvolution algorithm like CLEAN can deconvolve the dirty image, which is w -corrected.

w -stacking approximates the w -correction with the number of stacks it chooses. If it uses as many stacks as Visibilities, we end up with an exact w -corrections. In practice, many Visibilities have a similar w -term. Grouping them together in a stack is an efficient way of approximating the w -correction.

Backwards pass

If the number of stacks is equal the number of Visibilities, w -stacking performs an exact w -correction. In practice, the number of stacks is far smaller than the number of Visibilities, and each stack does one w -correction for similar Visibilities.

Inverse nuFFT with w -stacking: Visibilities get grouped into stacks by their w -term. Every visibility inside a stack has a similar w -term. The more stacks, the more accurately you can handle the w component. The figure displays the relation

The nuFFT gets called on each stack separately, and a constant w correction factor applied. Since w -term is constant, we can use the two dimensional nuFFT again. At the end, all the stacks are summed together to form the dirty image.

nuFFT with w -stacking: The inverse is done in a similar fashion. The sum cannot be re-separated, so the image gets copied into each stack. The constant w factor gets divided out. The nuFFT gets applied.

the runtime

(talking about more recent hybrid approach, limiting the total number of ? w -stacks?)

2.2 Self-Calibration

Complex gain term. Corrects amplitude and phase.

Traditional Calibration

A calibration source close by, with a known brightness. Phase and amplitude calibration was done before image reconstruction. For older interferometers, there was a very limited number of calibration terms to solve.

but again effects of wide field of view also increase the number of necessary calibration terms. The advent of self calibration, in which the image reconstruction was used to solve for both, the observed image and the calibration of the instrument.

Self calibration with the major cycle algorithm and a CLEAN deconvolution.

Initial phase calibration Shallow clean phase calibration deep clean phase and amplitude calibration deep clean reconstruction

3 Eliminating the Major Cycle

There is little research in exploring a different architecture for Compressed Sensing. The closest to this work was done by Pratley et al[10], which optimizes the non-uniform FFT in the context of compressed sensing. The Major cycle is still used. On the upside, progress in w-term accuracy and runtime.

There are ways to form a different architecture. However, the advent of self-calibration pushes a forward-backward kind of architecture on the reconstruction.

3.1 Separating the Major Cycle into two Optimization Problems

The major cycle reduces two errors simultaneously. The error introduced by the non-uniform FFT approximation, and the error introduced by the deconvolution approximation. In a sense, the major cycle is a simple optimization algorithm: It iteratively calls the non-uniform FFT and deconvolution approximations with the error from the last iteration, until the error is below a certain threshold.

$$FT : \underset{I_{dirty}}{\text{minimize}} \|V - A^{-1}I_{dirty}\|_2^2 \quad (3.1)$$

$$\text{deconvolution} : \underset{x}{\text{minimize}} \|I_{dirty} - x \star PSF\|_2^2 + \lambda \|P(x)\|_1 \quad (3.2)$$

The major cycle can also be separated into two objectives (3.1) and (3.2), and solved in sequence. The first objective (3.1) tries to find the best uniformly spaced image I_{dirty} matching the observation. A^{-1} represents the non-uniform inverse FFT approximation ($A^{-1}I_{dirty} \approx F^{-1}I_{dirty}$). When this is solved, the original Visibility measurements are not needed any more. Then, the second objective (3.2) searches for the deconvolved image x regularized by some prior $P()$.

By separating the two objectives, the hope is we can use specialized optimization algorithms that overall converge faster than several major cycles. Also, after the first objective was minimized, we can drop the Visibilities, which can be magnitudes larger than the image.

At first glance, this is a sensible idea, but the devil is in the objective (3.2). Namely in the PSF is not constant, it varies over the image. In the Major Cycle Architecture, a constant PSF^1 is sufficient: It only needs to approximate the deconvolution. The error introduced by a constant PSF can be reduced in the next Major Cycle. By eliminating the Major Cycle, we also eliminated the chance to approximate the deconvolution with a constant PSF . In the worst case, we would need to estimate the PSF for each pixel in the image.

In the worst case, these many PSF s may stop us from using this approach. It is plausible that can use fewer PSF s and be close enough. No known work done in the radio astronomy. The Major Cycle is ubiquitous. But it is plausible.

However, the problem is that the imaging algorithm should also be able to calibrate. With this approach, there is a strict one way flow from measurements to image. For calibration, a backwards flow from image to measurements is necessary.

For calibrated data, this approach may be potentially faster. but since MeerKAT will be difficult to calibrate correctly, this is unlikely.

Fourier Transform for PSF

¹CLEAN implementations in Radio Astronomy typically use a constant PSF . This is not strictly necessary, one can also implement CLEAN with a varying PSF .

3.2 Reconstruction on the Sphere

The signal naturally lives on the celestial sphere.

Not that much used in Radio Astronomy reconstructions. Work that uses to project on the sphere and looks at the reconstruction quality [11].

The wide field of view measurement equation (2.1) can be expanded into spherical harmonics.

[12] measurement equation with spherical harmonics. Essentially replaces the non-uniform FFT from the Major cycle with a new approach based on the fact that it has a solution to the Helmholtz equation. , see [12] for further detail. It can be used as a different way to arrive at a dirty image from wide field of view interferometers. It is useful in full sky image, because it fulfills the (property from paper) and the full sky dirty image does not have artifacts. It does not gain any complexity advantage, quite the contrary.

Even if the spherical harmonics transform would gain a speed advantage, used in the way of [12] it would still be the same major cycle architecture, just with spherical harmonics transform.

However there is the possibility to just do everything on the sphere.

Spherical haar wavelets used for simulating visibilities from a known image, essentially the opposite operation from imaging. [13]

based on the spherical harmonic transform. Doing a Spherical harmonic transform efficiently is still an active field of research [14]

transform . The wide field of view measurement equation (2.1) can be factorized into spherical harmonics. This means spherical harmonics are equivalent, calculating an image from the measurements with the three dimensional fourier transform is the same as using the spherical harmonics transform. [15]

[16]sampling theorem on a sphere

Spherical harmonics were researched in the context of Compressed Sensing image reconstructions (cite wiaux). They improved the quality of the image reconstruction, but no speed advantage.

The push on solving on the sphere directly. Using spherical haar wavelets, (cite) was able to speed up the simulation problem. Simulation tries to solve to problem of finding the measurements corresponding to a given image x . However, this has so far not been used for imaging.

3.3 Direct Fourier Transform

Hardy et al[17] handled the whole inverse Fourier Transform as an explicit matrix F^{-1} of size $M * N$. They dropped the non-uniform FFT and w -stacking approximations. This is trivially to distribute later. The only problem is of Course that for MeerKAT data sizes $M * N$ is too large (4gb of visibilities times 1 Billion Pixels).

However there is potential for improvement: Only a limited number of pixels in a reconstruction are not zero. We spent a lot of processing power on pixels which do not matter in the reconstruction. If we could predict which pixels are not zero, we would only need to calculate a subset of F^{-1} columns to reconstruct the image. The starlet transform[18] has a way to predict its non-zero components. We represent the image as a combination of starlets, and estimate which starlets are likely non-zero from the Visibilities directly.

To my knowledge, such an algorithm has never been tried out.

A proof-of-concept algorithm was developed that shows the prediction actually works in practice. It uses Coordinate Descent with the starlet transform, and only needs to calculate a subset of the matrix F^{-1} for a

reconstruction. The question remains how many columns of F^{-1} are needed, and is it more efficient than an algorithm using the major cycle architecture.

4 Compressed Sensing with direct Fourier Transform and Coordinate Descent

Instead of using the Major Cycle architecture and approximate the non-uniform FFT approximation, we directly use the Fourier transform matrix F . We show the principle of our approach and of Compressed Sensing reconstructions in general at a simplified example. Let us minimize the objective function (4.1).

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

The data term $\|V - F^{-1}x\|_2^2$ forces our image to be similar to the measurements, and the regularization term $\|x\|_1$ tells us how likely the current reconstruction is to be the true image. The parameter λ weights the trade-off between the data and the regularization term. With our term we assume our image contains only a limited number of non-zero pixels. When the instrument observes stars, they take up a single pixel in the image. In this case, our prior models the true image well, and the theory of compressed sensing states we are virtually guaranteed to find the truly observed image at the minimum of our objective function (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 operation². 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. Now when we iterate over all pixels several times, we eventually converge to a solution. 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[19]. Sadly, real-world objective functions for image reconstruction are often not strictly convex. Usually our image is constrained to have only positive pixels[11], which breaks the strictly convex property of

²The shrink operation reduces the magnitude of the pixel by λ . For example: Pixel $x = -12.4$, $\lambda = 0.6$. The new pixel value after shrinkage follows as $\text{shrink}(-12.4, 0.6) = -11.9$

the objective function. In our environment, the convergence guarantees of Coordinate Descent are not well understood in theory.

In practice, Coordinate Descent can be improved with heuristics. Since we assume our image contains only a few stars, a few non-zero pixels, the simple Coordinate Descent algorithm wastes resources checking if the pixel is still zero. A better scheme would be the active set heuristic[20]: 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 impossible in the context of MeerKAT. We have too many pixels and Visibilities, iterating over every pixel once is too expensive.

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

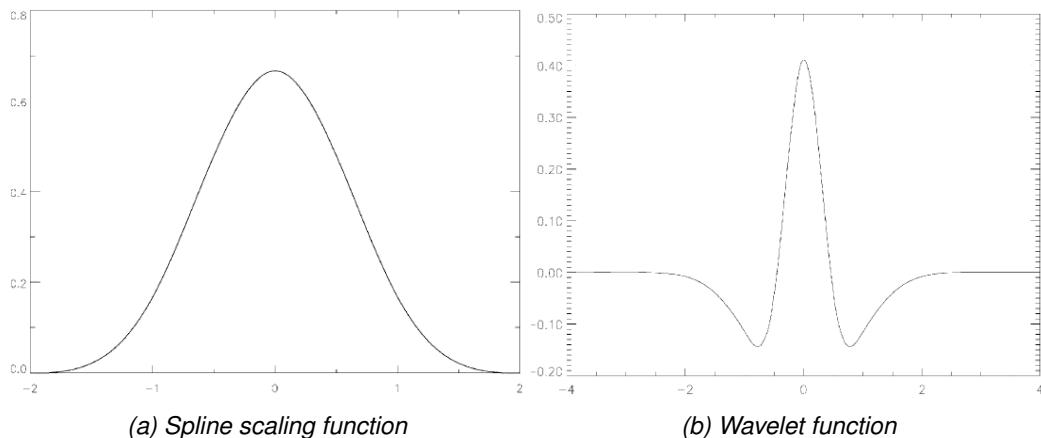


Figure 4: Scaling and wavelet function of the starlet wavelet. Source: [18]

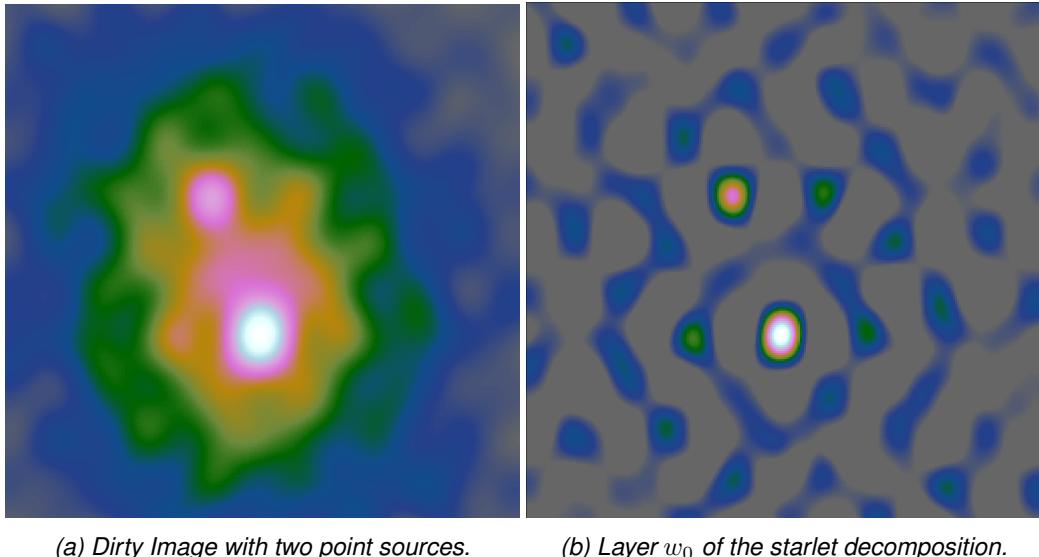
The starlet transform is an Isotropic Undecimated Wavelet Transform (IUWT) with the starlet wavelet shown in figure 4. As such, it defines two operations: The transformation from starlet into image space which is called synthesis, and the inverse which is called decomposition. The starlet space represents 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 largest 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, and how it is calculated from the image. B , and \mathbf{x} . $\mathbf{x} - c_0$ removes the larger structures from the image, leaving us with the smallest in w_0 . Does are not the starlet components, α_0 . However, we can estimate them with a shrink operation. we can then convolve α with the starlet function to get back to w_0 . In our implementation

In this case, we use it for estimating likely non-zero α . Figure 5 shows an example for the lowest starlet layer.



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

Figure 5: Dirty map and w_0 starlet layer. The right image shows the probability distribution for point sources in the dirty image.

The full decomposition is then just repeated for each intermediate layer w_i . (4.3) !!!!!A trous algorithm, size of the different S_i The last, heavily blurred c_J contains the structures which were too big to represent with the largest starlets, and is the last layer of the decomposition.

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

The synthesis transform however is easy.

$$\mathbf{x} = \mathbf{w}_0 + \mathbf{w}_1 + \dots + \mathbf{w}_{J-1} + \mathbf{c}_J \tag{4.4}$$

The starlet decomposition is over-complete.

The difference between α and w_i is not always important. For the SASIR algorithm, for FISTA, which is how [5] used the starlet transform for regularization, shrinkage on w_i without convolution.

For FISTA, this is irrelevant, but for Coordinate descent it is important. note on the shrinkage

the original starlet people [18] does not differentiate between α and w_i , and neither did [5]. For the Coordinate Descent implementation it is relevant for handling the positivity constraint.

4.1.1 Handling the positivity constraint

As mentioned in section 4, image reconstruction algorithms for radio astronomy typically constrain the image to be non-negative[11].

4.2 Coordinate Descent Implementation

We have the starlet regularization, let us put it into an algorithm. Start with the

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

Proof of concept implementation. There are many ways to improve the efficiency and the convergence speed. It is here to show the general if it is possible of the approach and if we can truly only calculate a subset of F^{-1} columns.

The objective function (4.5) minimizes the starlet components α . This objective leads again to a parabola in the one dimensional case, and we can analytically find the optimum for a single component when all others are fixed.

There are still a few problems to solve, like how to handle the matrix product $F^{-1}D$, With the active set heuristic, we iterate over a limited number of starlet components at a time.

The question remains, how the matrix product $F^{-1}D$ can be calculated efficiently. Since starlet transform is just made up of a couple of convolutions, we do not have to calculate the product $F^{-1}D$ explicitly. Convolutions in image space are multiplications in Fourier space. So for any starlet layer w_i or c_J , we can pre-calculate a transform vector. $w_2 = F^{-1}V * (\hat{S}_0\hat{S}_1 - \hat{S}_0\hat{S}_1\hat{S}_2)$

How many layers are needed. starlets rise in pixels with 2^J . Meaning if we want starlets over the whole image, the number of starlets rise logarithmically to the image dimensions.

How to iterate over the image in detail.

4.3 Iteration scheme

Many ways to iterate over the image. A simple scheme was used here.

Active set heuristic, find the starlets which are

```

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 with active set heuristic

sub-zero pixels

4.3.1 Non-uniform FFT approximations

Non uniform FFT to calculate the convolution KERNELS!!

Stupid approach with line search. Could be done more efficiently by using the histogram of the starlet level.

5 Test on simulated data

In this section, we test our new approach with Coordinate Descent on simulated MeerKAT data. We show that our approach does not need to calculate the whole Fourier Transform matrix F . Instead, we use a heuristic to only use relevant columns. 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, compared to CLEAN which overshoots the number by a factor of 1400. The total flux of the CLEAN reconstruction 6a overshoots the 3.9 figure by a factor of 1400. This gets obvious when we compare the intensity profile of CLEAN, Coordinate Descent and the ground truth in figure 7.

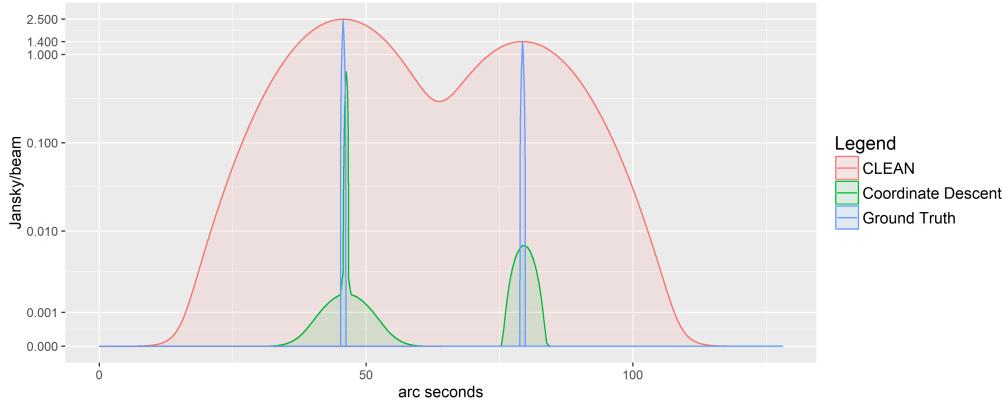


Figure 7: Intensity profile of the two point sources.

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. Coordinate Descent keeps the total flux in mind. The pixels in each area of the point sources sum up to the correct values of 2.5 and 1.4 respectively. However, note that Coordinate Descent in figure 7 seems to have 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

Dataset of three gaussian extended emissions and sixteen point sources at varying intensities.

Parameters of Coordinate Descent

super resolution of CD. However, it did not find all point sources Lets look closer at the flux reconstruction of extended emissions

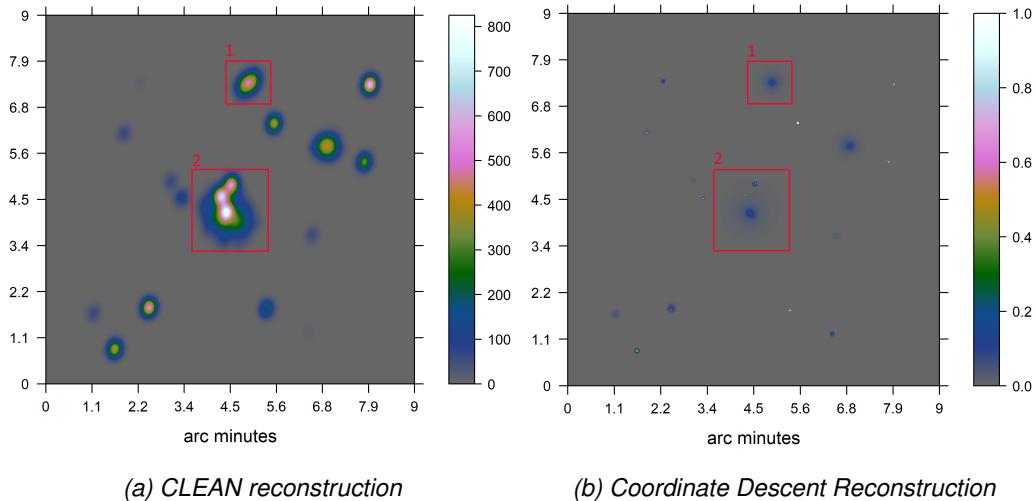


Figure 8: Reconstruction on mixed sources

Question of Flux reconstruction

λ for different starlet layers like in [5]

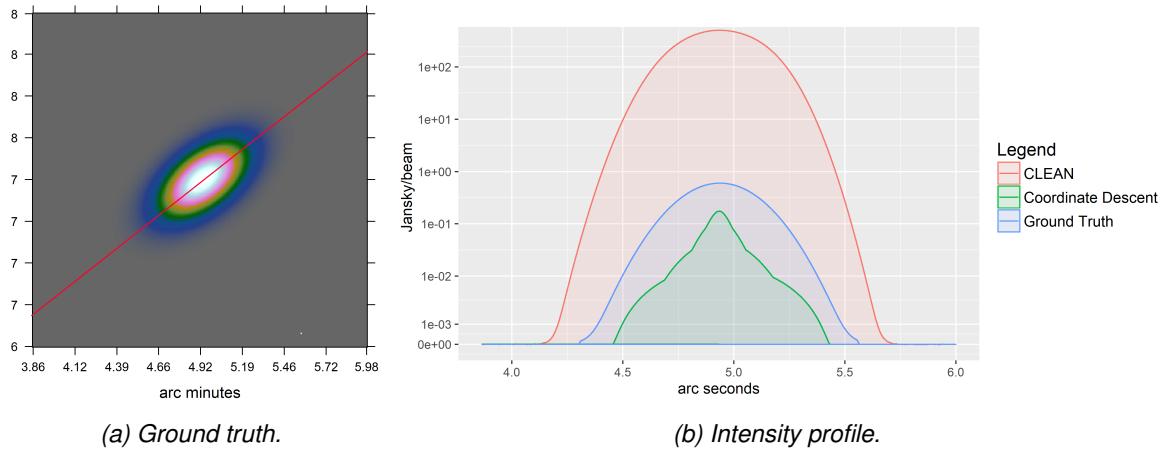


Figure 9: Intensity profile of region 1.

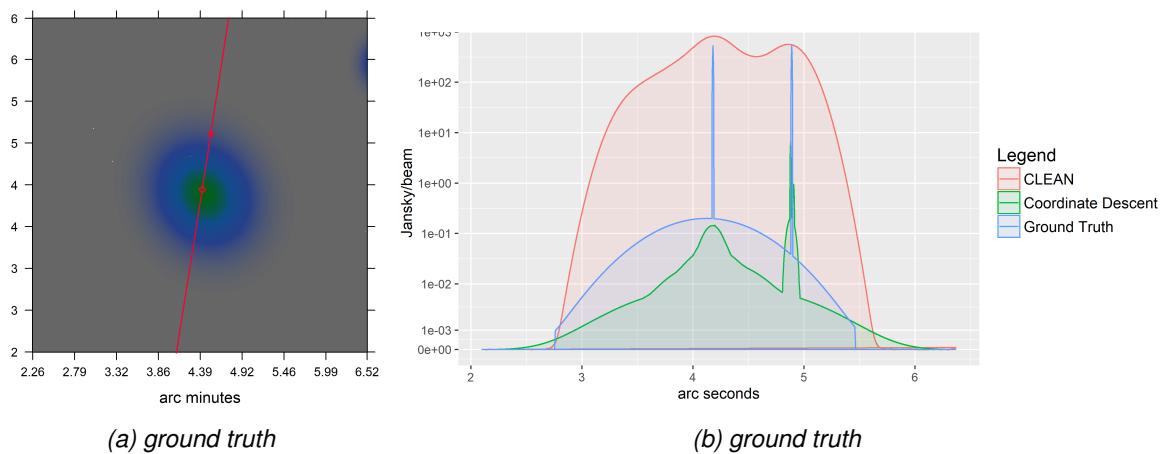


Figure 10: Intensity profile of region 2.

Coordinate Descent did not reconstruct all point sources. How many starlets are non-zero is the major point for runtime. It depends on how many areas of the image are non-zero. Starlet has a representation for extended emission, how many starlets are needed for modelling is hard.

Runtime problems 244 non-zero starlets,

show the areas

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 11. 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:

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

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.

$$\text{non-uniform FFT : } M + 2N * ld(2N)$$

$$\text{non-uniform FFT with } w\text{-stacking : } M + W * (2N * ld(2N) + 2N) + N * ld(N)$$

$$I_{WSCLEAN} \text{ deconvolutions : } I_{WSCLEAN} * 2N$$

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 certain number of iterations.

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

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 11, 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

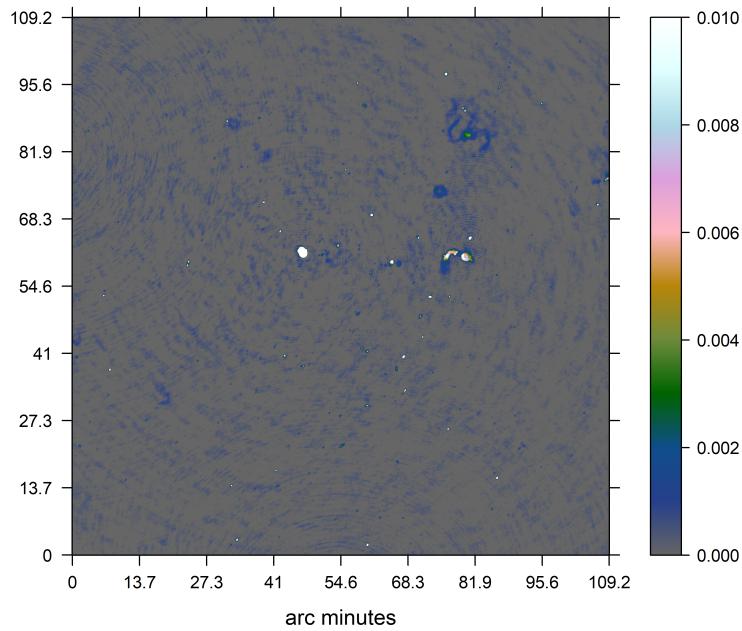


Figure 11: WSCLEAN Reconstruction of the MeerKAT observation.

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 11 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 1.9 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.[9] 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 1.2 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 terabytes³ in our case. If not, the columns have to be re-calculated on the fly, increasing the runtime costs by factors.

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 $I_{CD} * [S * 4N + \dots]$. By using the Major Cycle architecture, Coordinate Descent can afford more iterations and more non-zero components in the image for the same runtime complexity. 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.

³If 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

In this project, we created a new architecture for Compressed Sensing reconstructions. Instead of using the non-uniform FFT approximation, we directly use the relevant columns of the Fourier transform matrix. We showed that the likely-relevant columns can be retrieved with a heuristic, and we can reconstruct images without ever calculating the whole Fourier matrix. This architecture naturally extends to the wide field of view measurement equation (2.1). It can handle the w -term explicitly and does not need any approximations.

We created a proof-of-concept algorithm in this architecture, with Coordinate Descent as the optimizer and starlets as regularization. We exploited the starlet for a greedy heuristic, locating the likely-relevant columns of the Fourier transform matrix. The same heuristic can be used with other regularizations. It locates the likely location of Gaussian-like objects at different scales. As such the same heuristic could be used for a Gaussian mixture model. At this point, it is unclear if the heuristic will eventually converge to the true optimum in theory. In practice, we showed super-resolution performance of Coordinate Descent on simulated MeerKAT data together with accurate total flux modelling compared to CLEAN.

This is a similar result to other Major Cycle compressed sensing algorithms, which have also shown super-resolved reconstructions compared to CLEAN on real-world observations[6][5], but also at a higher runtime cost. The main goal of this project was to find a new architecture which reduces the runtime cost of Compressed Sensing algorithms for large scale reconstructions. In the context of MeerKAT, the estimated runtime costs of CLEAN were still half of the best case estimate of our Coordinate Descent algorithm. Ironically, the runtime costs of Coordinate Descent can be improved by using the Major Cycle architecture instead of the direct Fourier transform and potentially reduce its memory requirement as well.

That does not mean the new architecture cannot beat CLEAN or the Major Cycle. Our architecture scales independently of the image size. The cost effectiveness of the two architectures boils down to the ratio between Visibilities and pixels. For MeerKAT reconstructions, the ratio is skewed towards Visibilities, and the Major Cycle is more cost effective. But if we increase the number of pixels in the reconstruction, we increase the cost-effectiveness of our direct Fourier transform architecture.

Increasing the number of pixels is only useful to a limit. Decreasing the number of Visibilities however, might yield better results: The large number of Visibilities are virtually guaranteed to have redundant information. A portion of the total runtime is spent on Visibilities, which do not contain any new information for the image. One improvement may be to sub-sample the Visibilities and reconstruct an approximate image from a fraction of the dataset. In this environment, our new architecture may become viable for reducing runtime costs. Indeed, the direct Fourier transform may not be cost effective partly because it does not use approximations. Our architecture calculates the Fourier transform up to the limit of floating-point precision. The Major Cycle on the other hand, uses the non-uniform FFT approximation and stops at the noise limit of the measurements. For our new architecture, introducing back approximations may be the key to reduce runtime costs.

In the future MeerKAT is scheduled to be expanded, posing the reconstruction problem on an even larger scale. Sooner or later, the reconstruction has to be done over a distributed computing infrastructure. In the Major Cycle, single steps like the w -correction lend themselves to distributed computing. But designing a Major Cycle algorithm which is highly distributable is an open problem. The simplified architecture of our direct Fourier transform together with Coordinate Descent may have an edge in distributed computing. Coordinate Descent can converge, even when the descent steps are distributed. As it exists in this project, the memory and runtime costs of our approach are too high to be relevant for MeerKAT reconstructions.

However, if we can decrease the memory and runtime costs with the right approximations, while keeping the distribution properties of Coordinate Descent intact, our approach may become interesting for distributed image reconstructions.

7 Compressed Sensing reconstructions in the MeerKAT era

MeerkAT poses a large scale reconstruction problem. Compressed Sensing approaches exist, they use the major cycle, but they all are slower to compute than CLEAN, for which the major cycle was developed.

It uses the non-uniform FFT to approximate the Fourier transformation of non-uniformly sampled Visibilities.

This project set out to explore different architectures for Compressed Sensing Reconstructions. A different way of getting from Visibilities to image.

Three different possibilities, with calculating the non-uniform FFT once, using spherical harmonics, or calculating only columns of the Fourier transform matrix

For the former, we created a novel approach which does not need the major cycle. We need a transformation, where we can both estimate which components are likely to be non-zero, and which use few non-zero components to represent an image.

This project used starlets, which were developed for astronomical images. Their solution is not sparse enough to scale to MeerKAT problems.

This leaves two other ways, splitting the major cycle falls off if we consider self-calibration

Spherical harmonics are the last way, Compressed Sensing algorithms have used it to gain another improvement in reconstruction quality. At this point, there is no Compressed Sensing algorithm which used the sphere for speed up.

here I see a possibility. The downside is it leads to a new measurement equation. It did not have as much time to formalize corrupting effects like antenna beam pattern, ionosphere and so on.

There is yet an

Still an open question

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] Jasleen Birdi, Audrey Repetti, and Yves Wiaux. Sparse interferometric stokes imaging under polarization constraint (polarized sara). *arXiv preprint arXiv:1801.02417*, 2018.
- [8] 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.
- [9] 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*, 2018.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] JD McEwen and AMM Scaife. Simulating full-sky interferometric observations. *Monthly Notices of the Royal Astronomical Society*, 389(3):1163–1178, 2008.
- [14] Nathanaël Schaeffer. Efficient spherical harmonic transforms aimed at pseudospectral numerical simulations. *Geochemistry, Geophysics, Geosystems*, 14(3):751–758, 2013.
- [15] J Richard Shaw, Kris Sigurdson, Ue-Li Pen, Albert Stebbins, and Michael Sitwell. All-sky interferometry with spherical harmonic transit telescopes. *The Astrophysical Journal*, 781(2):57, 2014.
- [16] Jason D McEwen, Gilles Puy, Jean-Philippe Thiran, Pierre Vanderghenst, Dimitri Van De Ville, and Yves Wiaux. Sparse image reconstruction on the sphere: implications of a new sampling theorem. *IEEE Transactions on image processing*, 22(6):2275–2285, 2013.

- [17] Stephen J Hardy. Direct deconvolution of radio synthesis images using l1 minimisation. *Astronomy & Astrophysics*, 557:A134, 2013.
- [18] Jean-Luc Starck, Fionn Murtagh, and Mario Bertero. Starlet transform in astronomical data processing. *Handbook of Mathematical Methods in Imaging*, pages 2053–2098, 2015.
- [19] 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.
- [20] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

List of Figures

1	The Major Cycle Framework	2
2	Celestial sphere distortion on simulated data. Source: [8]	5
3	The Major Cycle Framework	5
4	Scaling and wavelet function of the starlet wavelet. Source: [18]	11
5	Dirty map and w_0 starlet layer. The right image shows the probability distribution for point sources in the dirty image.	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	WSCLEAN Reconstruction of the MeerKAT observation.	20

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 23, 2019

Jonas Schwammburger