SKA AND VLA PRIMARY BEAM PATTERN SIMULATIONS WITH MEQTREES AND CASA

Mark Yashar and Athol Kemball, UIUC

February 2012

ABSTRACT

We describe the details and results of Meqtrees and CASA Square Kilometer Array (SKA) observation and imaging simulations we have carried out involving the use of the Cortes primary beam pattern model to better understand and assess the effect of direction-dependent calibration errors (including primary beam effects and pointing errors) on SKA image fidelity and dynamic range. We have also carried out VLA primary beam simulations with CASA to better understand the implementation of the A-Projection algorithm within the CASA software package and to test and assess its performance. Finally, we have attempted to make some improvements in the implementation of A-Projection in CASA to correct for the effects of the VLA primary beam on simulated VLA visibility data.

1.0 INTRODUCTION

Astronomical observations made with radio interferometers suffer from direction-dependent calibration effects such as primary beam effects. These direction-dependent effects are expected to limit observations with existing as well as next generation radio interferometers presently under development such as the SKA. Wide-field, high dynamic imaging with the SKA will require such direction-dependent calibration effects to be accounted and corrected for within the radio interferometric measurement equation (RIME) or full-sky Jones formalism [1,2]. A number of algorithms have been developed in recent years, including the A-Projection algorithm [3], to incorporate and correct for direction-dependent gains during the imaging and deconvolution process. These algorithms, however, need to be further tested to assess their performance against different antenna and interferometer design parameters (e.g., antenna diameter, mount type, feed design parameters) , as well as different computing hardware and software architectures, and further developed and improved as necessary [1]. The construction of radio synthesis telescopes such as SKA brings about a number of new technical challenges including a required increase in algorithmic performance in terms of sensitivity and dynamic range, as well as the need for algorithms adapted to high-performance computing [4,5].

In this document we describe a number of Meqtrees [6] and CASA (http://casa.nrao.edu) simulations we have carried out involving a SKA antenna primary beam pattern model to assess the effect of this beam model on the fidelity and dynamic range of simulated SKA images. The eventual goal of this project is to employ the A-Projection algorithm within the CASA software package, which (like Meqtrees) implements the RIME formalism [7], to correct for the effects of an SKA primary beam model (the Cortes beam model in this case) on simulated SKA visibilities, and then to evaluate the r.m.s. errors, mean pixel brightness values, and dynamic ranges of the resulting simulated images.

As an intermediate step toward reaching this goal, we also describe CASA VLA simulations we have carried out to better understand the implementation of A-Projection within CASA and to test and assess A-Projection's performance. We then describe our efforts to improve the performance of A-Projection in CASA to correct for the effects of the VLA primary beam on simulated VLA observations. Finally, we outline plans and suggestions for future work.

2.0 SKA CORTES BEAM PATTERN SIMULATIONS WITH CASA AND MEQTREES

We carried out a number of SKA observation and imaging simulations with the CASA (stable version 3.02) software package (http://casa.nrao.edu/) and the Meqtrees software package (http://www.astron.nl/meqwiki) with antennas placed in a uniform random distribution within a log-spiral configuration with number of antennas N_a = 50,75,100,150,175 and maximum baseline length B_{max}~35 km. Figure 1 shows an example plot of the logarithmic spiral configuration for the case of N_a = 150.

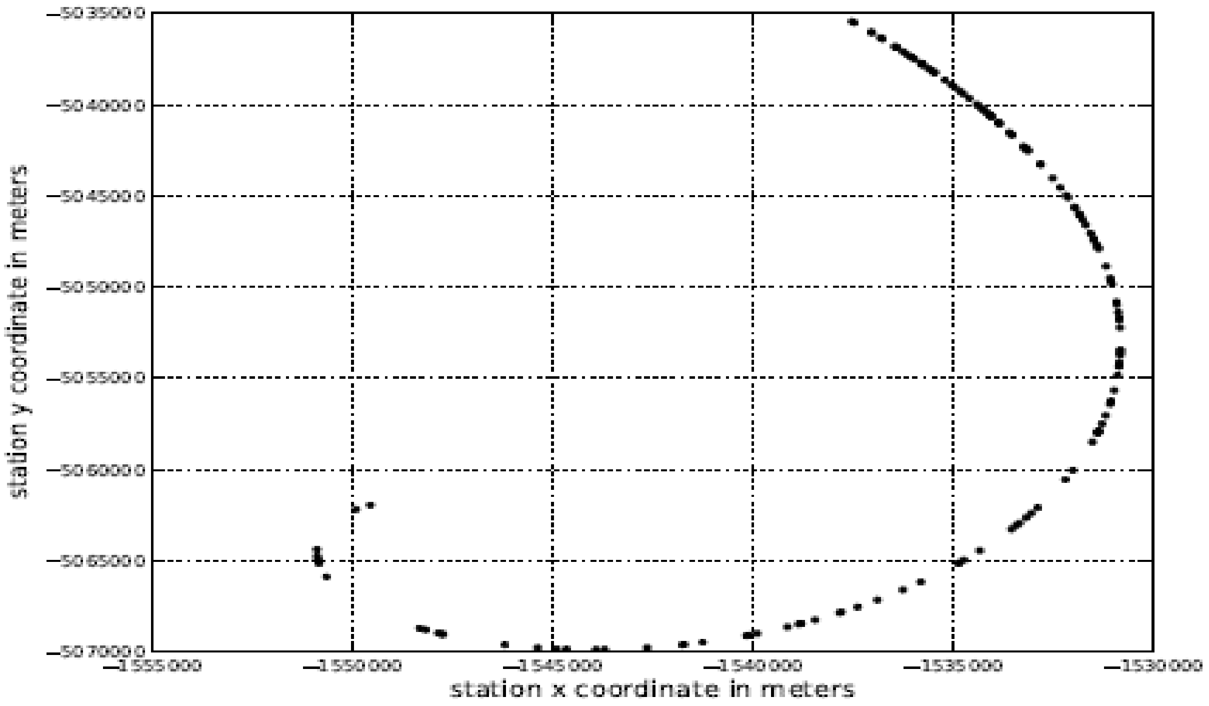


Figure 1: Plot of Logarithmic Spiral antenna configuration pattern with N_a = 150 and Bmax ~35km. Antennas have been placed in a uniform random distribution within the log-spiral configuration. The (x,y) antenna location coordinates are in Earth-centered Earth-fixed cartesian coordinates (whereby the coordinate system rotates with the Earth and has origin at the Earth's center) centered on the location of the center of the VLA.

The simulations were carried out utilizing python CASA scripts and Meqtrees Tree Definition Language (TDL) python scripts, along with SKA Cortes antenna beam pattern model data, provided by Tony Willis, in combination with our specified SKA antenna positions (log-spiral configuration). We generated our

SKA antenna positions (e.g., Figure 1) from a Python script we wrote which placed N_a antennas randomly within a logarithmic spiral configuration with a maximum baseline length B_{max} of about 35 kilometers with the array center located at the same latitude and longitude as the array center of the VLA. The output antenna locations (X,Y,Z in Earth-Centered Earth Fixed Coordinates (ECEF)) were written to a text file which includes the following columns:

```
Antenna ID#    Array Name    X    Y    Z    axis type    diam
```

where ‘Antenna ID#’ is the identification number for an SKA antenna (e.g., SKA12) ,’Array Name’ is the name of the array (e.g., SKA), X,Y, and Z are the earth-centered-earth-fixed Cartesian coordinates of each antenna location, ‘Axis type’ is the mount type for each antenna in the array (ALT-AZ for SKA), and ‘diam’ is the dish diameter for each antenna in the array (e.g., 12.0 meters for SKA). So, for example, the first few entries of a text file containing the locations of antennas in a , e.g., random log-spiral configuration would look like:

```
SKA0   SKA   -1535957.36043   -5037516.88392   3547260.06198   ALT-AZ   12

SKA1   SKA   -1531383.4966   -5047820.59898   3541872.92399   ALT-AZ   12

SKA2   SKA   -1531072.10716   -5049522.01737   3549655.77271   ALT-AZ   12

SKA3   SKA   -1545368.7624   -5069781.52277   3542748.83373   ALT-AZ   12
```

We were then able to modify Tony Willis' Meqtrees scripts in such a way that they could be used as 'beam module' scripts for use with the Meqtrees 'Station Jones Module' and 'Sky Jones Module' contracts (http://www.astron.nl/meqwiki). The SKA Cortes beam pattern model is described in [8,9]. Figures 2-4 shows plots of the Cortes beam radiation pattern generated from the beam pattern model data provided by Tony Willis. See Appendix 1 for 3-dimensional scatter plots of the Cortes beam pattern.The full width half maximum (FWHM) of the Cortes primary beam used in the simulations was approximately 75 arcminutes.

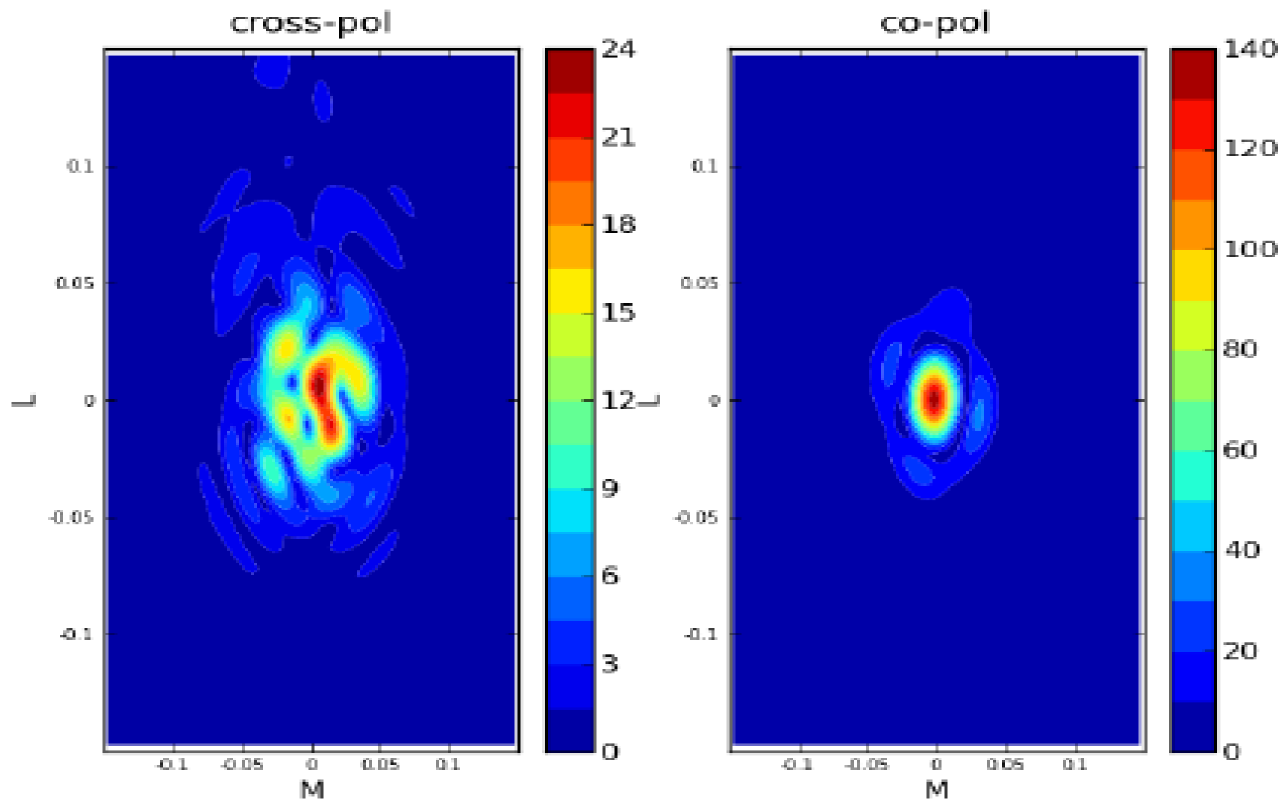


Figure 2: The full Cortes beam radiation pattern of an offset Gregorian reflector optics design for one antenna dish with a 12 meter aperture diameter and a 3 meter subreflector size. The subreflector illumination angle is 42 degrees and the observing frequency is 1.4 GHz. L,M coordinates are in radians and L=M=0 at beam peak. Left panel: cross-polarization. Right panel: co-polariztion.

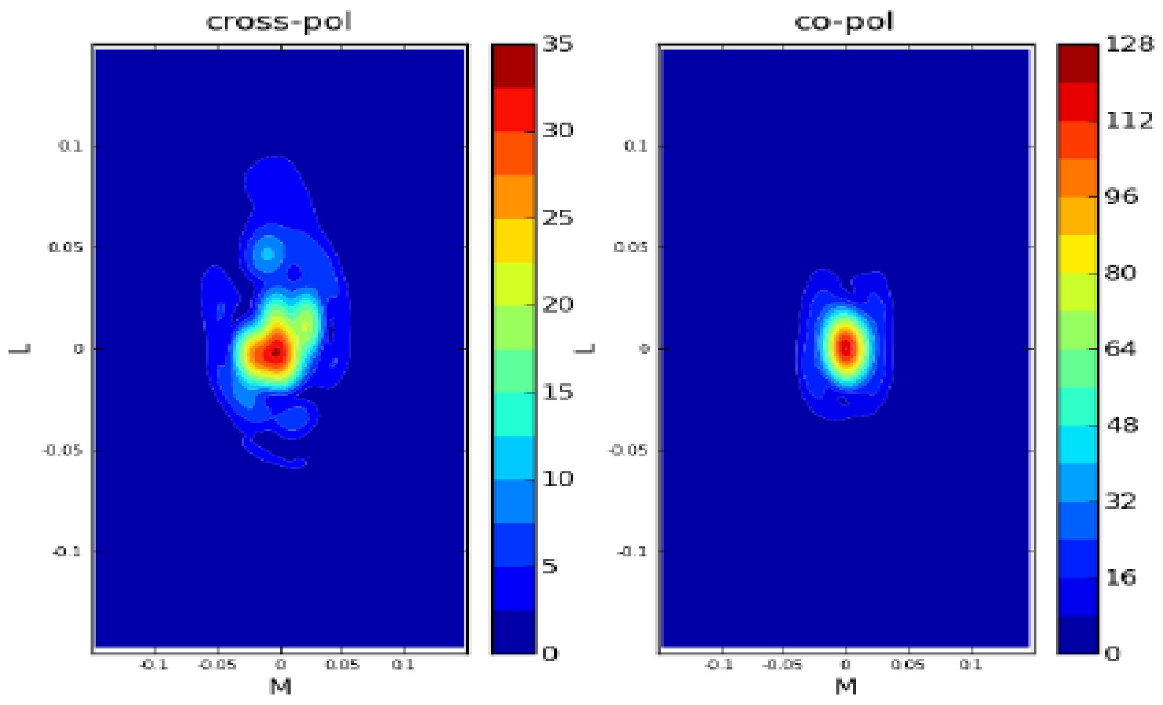


Figure 3: Full Cortes beam radiation pattern of offset Gregorian reflector optics design with Mizusawa-Mizugutch conditions for one antenna dish with 12 meter aperture diameter and 3 meter subreflector size; subreflector illumination angle is 67 degrees; observing frequency is 1.4 GHz. L,M coordinates in radians and L=M=0 at beam peak. Left panel: cross polarization, right panel: co-polariztion.

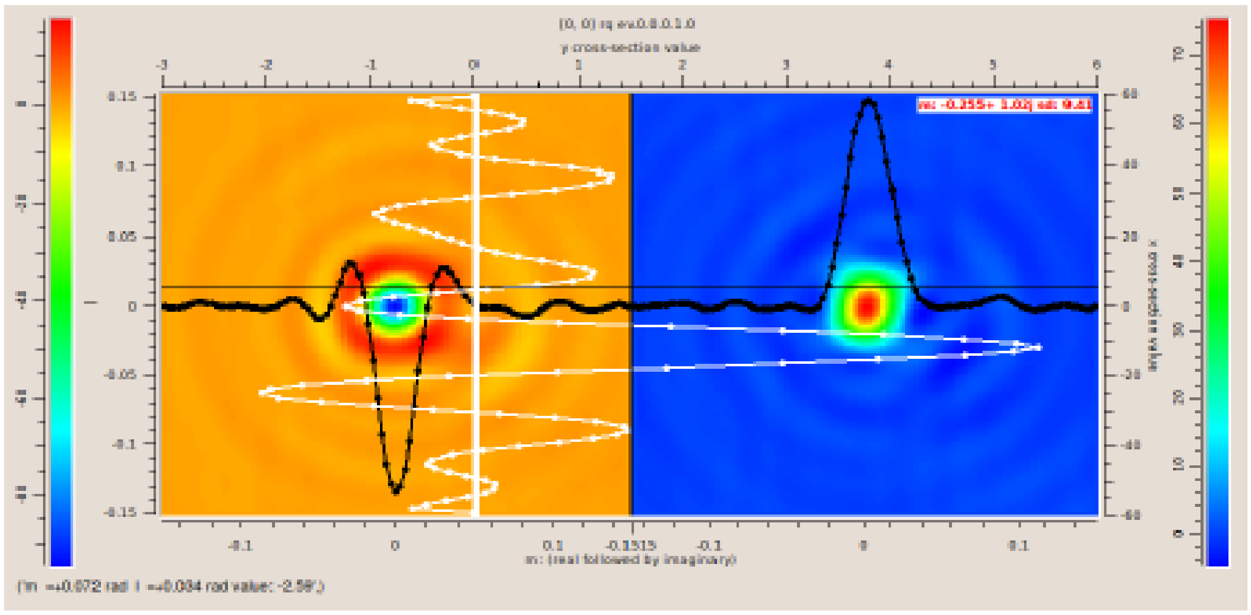


Figure 4: Meqtrees-generated contour plot of real (left panel) and imaginary (right panel) parts of total power response of Cortes primary beam pattern model (Figs. 2 and 3) for an interformeter pair. Here, Stokes I = Stokes Q and Stokes U = Stokes V = 0. Black and white curves and points are for x and y cross section values.

In some simulations, a pointing error model (provided by Tony Willis) with pointing errors l_{offset}=0.00172 radians = 5.919 arcminutes and m_{offset} = 0.0004 radians = 1.416 arcminutes was also used together with the Cortes beam pattern model. In all simulations discussed here, the simulated SKA observations (and the corresponding UV tracks) were generated with CASA with the following observation parameters:

Location of telescope:	Same as VLA
Field center RA:	0h0m0.1s
Field center DEC:	28d0m0s
Frequency at lower edge of band:	1400MHz
Number of Channels:	64
Channel increment:	5.0MHz
Dish diameter in meters:	12.0
Number of dishes in the array:	50, 75, 100, 150, 175
Stokes parameters:	XX XY YX YY
Integration period:	60s
Reference time:	2010/01/01
Start of observation in seconds relative to reference time:	0.0
Length of observation (in seconds):	5400.0
(Added) noise:	0.0Jy
Field of view in acrmin:	150
Number of sources to observe:	2

An ‘empty’ Measurement Set (MS) was then generated with CASA using the simulated observation. The empty MS consisted of antenna information, a central pointing direction, frequency channel information and the empty UVW points. Then, Meqtrees generated visibilities based on a 2-point source sky model that we provided with sources separated by approximately 1 FWHM (~ 75 arcmin) with each source having a brightness of 1 Jy. These visibilities filled up one or more data columns of the MS. Meqtrees was also used to corrupt the simulated visibilities with the Cortes beam model with and without the pointing error model applied. Finally, CASA was used to generate image maps from the corrupted visibilities (or un-corrupted visibilities, depending on whether the visibilities were corrupted with the Cortes beam model for a particular simulation). The w-projection algorithm [10] (with 128 w-projection planes) was also used in the imaging process. Deconvolution was not carried out in these simulations due to technical issues. The imaging simulations included the following three cases:

(1) CASA-generated dirty images of the two-source sky model for N_a = 50,75,100,150,175 with the visibilities not corrupted by the Cortes beam model.
(2) CASA-generated dirty images of the two-source sky model for Na = 50,75,100,150,175 with the visibilities corrupted by the Cortes beam model without pointing errors.
(3) CASA-generated dirty images of the two-source sky model for Na = 50,75,100,150,175 with the visibilities corrupted by the Cortes beam model with pointing error model utilized.

We also generated ‘difference’ images that resulted from subtracting images made from visibilities corrupted by the Cortes beam model with no pointing errors from images made from visibilities corrupted by the Cortes beam model and pointing errors utilized in order to assess the effect of the pointing errors on the image fidelity and imagin dynamic range. We also generated differenced images between an image with visibilities corrupted by the Cortes beam model (without pointing errors) and an image with visibilities not corrupted by the Cortes beam model. Image statistics were obtained for each image including means, medians, root mean squares, standard deviations, imaging dynamic range estimates, and ranges of pixel brightness values. The statistical measures for each image were obtained within a particular bounding box region between but not including the two sources, as well as within small box regions around each of the two sources. Imaging dynamic range estimates were calculated for each image by taking the ratio of the brightest (largest) positive pixel brightness value in all of the image to the root mean square of pixel brightness values in a fixed region (same region where other statistical measures were obtained) between but not including the two sources. Corresponding histogram line plots of counts vs. pixel brightness values were also generated for each image. The image maps and histogram line plots are too numerous to include in this document but are available upon request. In all simulations discussed here we label the source at field center as 'source 1' and the source displaced 75 arcmin from field center as 'source 2'. A few examples (for the case of N_a=150) are included in Figures 5-11 below.

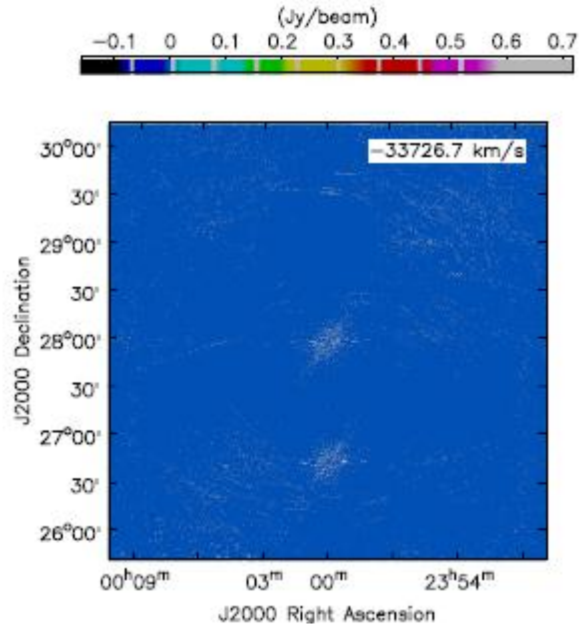
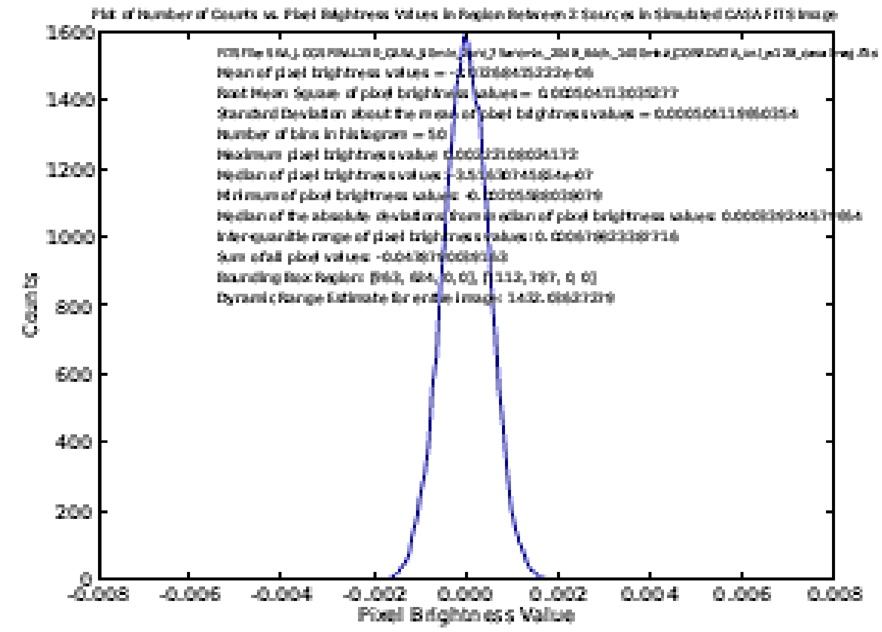


Figure 5: CASA SKA simulation and CASA-generated dirty image of two 1 Jy point sources with 75’ separation located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s , without Cortes beam model corruption of visibilities. Simulation: Na = 150; image displayed and brightness-contrast colormap adjustments with casaviewer; colormap used: ’isophotes’.



```
FITS File: [illegible]
Mean of pixel brightness values = -2.[illegible]e-06
Root Mean Square of pixel brightness values = 0.000504713035277
Standard Deviation about the mean of pixel brightness values = 0.000504111986035 4
Number of bins in histogram = 50
Maximum pixel brightness value: 0.00222710860417 2
Median of pixel brightness values: -3.51163074585 4e-07
Minimum of pixel brightness values: -0.00205386003076
Median of the absolute deviations from median of pixel brightness values: 0.000330245719864
Inter-quartile range of pixel brightness values: 0.000678023308716
Sum of all pixel values: -0.0438719006[illegible]
Bounding Box Region: [963, 634, 0, 0], [1112, 787, 0, 0]
Dynamic Range Estimate for entire image: 1432.08827219
```

Figure 6: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [963,634,0,0], top right corner = [1112,787,0,0]) between but

not including the two point sources in Fig. 5. Statistical measures calculated within the bounding box region are included in the plot. The dynamic range estimate is for the entire image and is taken as the ratio of the brightest (largest) positive pixel brightness value in the image to the rms of pixel brightness values in the bounding box region.

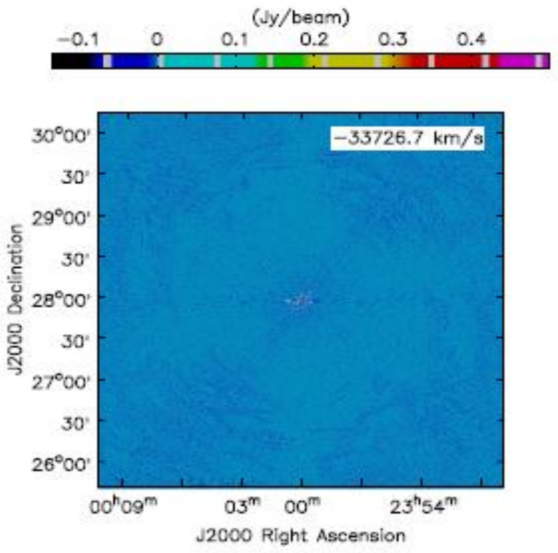
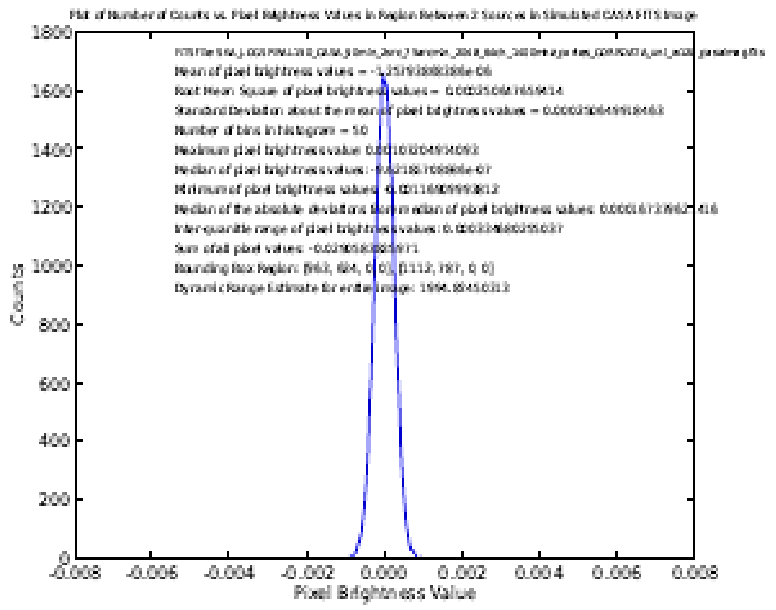


Figure 7: CASA-Meqtrees SKA simulation and CASA-generated dirty image of two 1 Jy point sources with 75' separation located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s, with visibilities corrupted by Cortes beam model (using Meqtrees) without pointing errors. Simulation: Na = 150; image displayed and brightness-contrast colormap adjustments with casaviewer; colormap used: 'isophotes'.



```
FITS file [illegible]
Mean of pixel brightness values = -1.25793088388e-06
Root Mean Square of pixel brightness values = 0.00025086765914
Standard Deviation about the mean of pixel brightness values = 0.000250649918463
Number of bins in histogram = 50
Maximum pixel brightness value: 0.00103204934093
Median of pixel brightness values: -9.62188708886e-07
Minimum of pixel brightness values: -0.00116809998812
Median of the absolute deviations from median of pixel brightness values: 0.00016733962[illegible]416
Inter-quartile range of pixel brightness values: 0.000334880235037
Sum of all pixel values: -0.0280588[illegible]971
Bounding Box Region: [963, 634, 0, 0], [1112, 787, 0, 0]
Dynamic Range Estimate for entire image: 1996.8845[illegible]0313
```

Figure 8: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [963,634,0,0], top right corner = [1112,787,0,0]) between but not including the two point sources in Fig. 7. Statistical measures calculated within the bounding box region are included in the plot.The dynamic range estimate is for the entire image and is taken as the ratio of the brightest (largest) positive pixel brightness value in the image to the rms of pixel brightness values in the bounding box region.

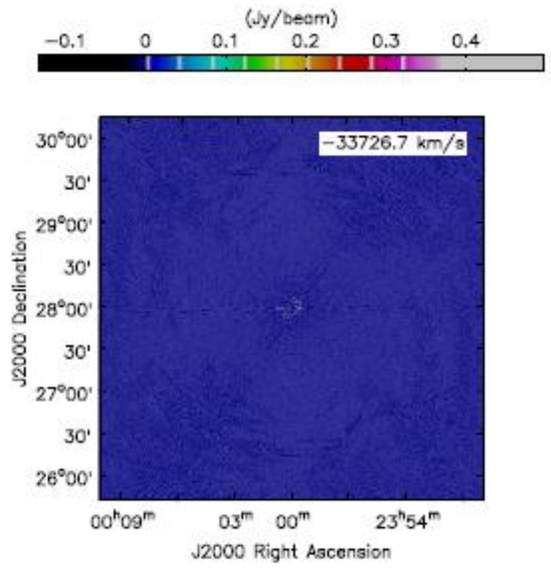


Figure 9: CASA-Meqtrees SKA simulation and CASA-generated dirty image of two 1 Jy point sources with 75’ separation located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s , with visibilities corrupted by Cortes beam model (using Meqtrees) with pointing errors (loffset = 0.00172 rad = 5.919’, moffset = 0.0004 rad = 1.416’). Simulation: Na = 150; image displayed and brightness contrast colormap adjustments with casaviewer; colormap used: ’isophotes’.

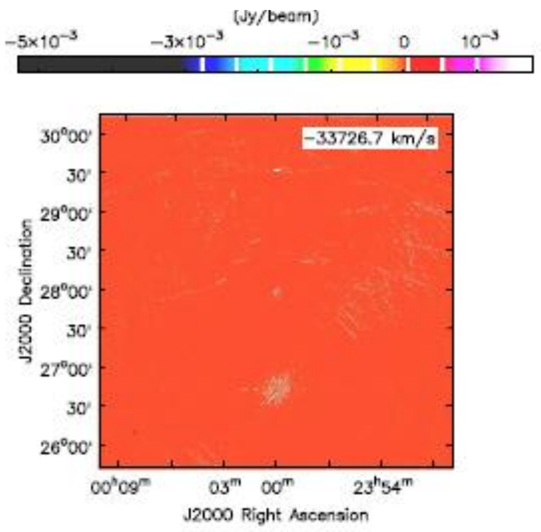
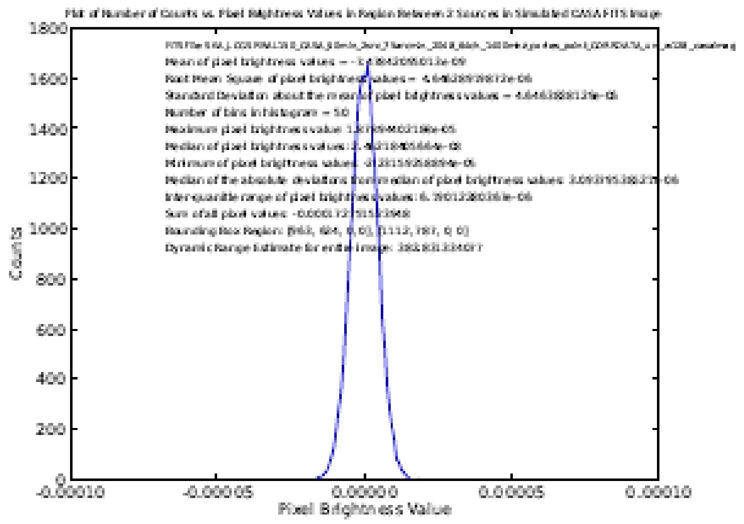


Figure 10: CASA-Meqtrees SKA simulation and CASA-generated dirty image resulting
from subtracting image from CASA-Meqtrees simulation with visibilities corrupted by
Cortes beam model and no pointing errors from image resulting from CASA-Meqtrees simulation with visibilities
corrupted by Cortes beam model and pointing errors. Simulation:
Na = 150, etc. (same as in previous.)



```
FITS File: [illegible]
Mean of pixel brightness values = -1.[illegible]e-09
Root Mean Square of pixel brightness values = 4.6463899872e-06
Standard Deviation about the mean of pixel brightness values = 4.6463888129e-06
Number of bins in histogram = 50
Maximum pixel brightness value: 1.87994602186e-05
Median of pixel brightness values: 2.46218405664e-08
Minimum of pixel brightness values: -2.22155926889e-05
Median of the absolute deviations from median of pixel brightness values: 3.09319538217e-06
Interquartile range of pixel brightness values: 6.19012280369e-06
Sum of all pixel values: -0.000172761512948
Bounding Box Region: [963, 634, 0, 0], [1112, 787, 0, 0]
Dynamic Range Estimate for entire image: 382.831334077
```

Figure 11: Line plot of counts vs. pixel brightness value for a specified bounding box
region (bottom left corner = [963,634,0,0], top right corner = [1112,787,0,0]) between but
not including the two point sources in Fig. 10. Statistical measures calculated within the
bounding box region are included in the plot. The dynamic range estimate is for the entire
image and is taken as the ratio of the brightest (largest) pixel brightness value in the image to the rms of
pixel brightness values in the bounding box region.

Table 1: Statistical results of pixel brightness values of FITS files of CASA-Meqtrees simulated dirty images of two 1Jy point sources separated by 75' with and without visibilities corrupted by the Cortes beam model (including with the pointing error model), and the difference between images with visibilities corrupted by the Cortes beam model with pointing errors and visibilities corrupted with Cortes beam model without pointing errors (signified as 'diff1'). Also included are the results for differenced images generated between an image with visibilities corrupted by the Cortes beam model (without pointing errors) and an image with visibilities not corrupted by the Cortes beam model (signified as 'diff2'). Note that the pointing error model, signified by 'poi1', includes loffset = 0.00172179 rad = 5.919 arcmin., moffset = 0.00041211 rad = 1.416 arcmin. Statistical results (in columns 2-5) were obtained within a particular bounding box region between but not including the two sources. The dynamic range estimates (D.R.) was obtained for the entire image, and the pixel range is the minimum and maximum pixel brightness values for all of the image.

N_a	**mean**	**median**	**r.m.s.**	σ	**D.R. (all)**	**pixel range (all)**
50	-1.42306e-06	4.07206e-06	0.00144768	0.0014477	691.338	-0.475156, 1.00084
50 (cortes)	3.34910e-07	1.59529e-06	0.000505216	0.000505227	989.725	-0.219612, 0.500025
50 (cortes,poi1)	3.51602e-07	2.41113e-06	0.000503330	0.000503340	989.837	-0.218816, 0.498214
50 (diff1)	1.669e-08	1.11758e-08	8.24857e-06	8.24873e-06	415.434	-0.00505518, 0.0034267
50 (diff2)	1.75797e-06	-1.35743e-05	0.0011601	0.0011601	422.889	-0.647514, 0.490594
75	1.1146e-05	5.78874e-06	0.000926000	0.000925953	840.981	-0.230988, 0.778749
75 (cortes)	4.92254e-06	2.46823e-07	0.000501860	0.000501847	996.296	-0.124897, 0.500002
75 (cortes,poi1)	4.92109e-06	-8.41526e-08	0.000501110	0.000501097	996.360	-0.12461, 0.499286
75 (diff1)	-1.45186e-09	1.4784e-08	5.19595e-06	5.19606e-06	260.140	-0.00210462, 0.0013516
75 (diff2)	-6.22421e-06	-6.61911e-06	0.000774828	0.000774819	268.275	-0.409090, 0.20786
100	-2.16327e-06	-4.35477e-06	0.00102543	0.00102545	975.332	-0.46253, 1.00014
100 (cortes)	-2.85182e-07	-2.924e-06	0.000323241	0.000323248	1546.8	-0.122883, 0.499998
100 (cortes,poi1)	-2.58316e-07	-2.63558e-06	0.00032188	0.000321890	1547.75	-0.122435, 0.498196
100 (diff1)	2.68662e-08	-9.89530e-10	5.83344e-06	5.83351e-06	499.225	-0.00514587, 0.00291220
100 (diff2)	1.87809e-06	8.73696e-08	0.000851987	0.000852003	561.082	-0.697748, 0.478034
150	-2.07268e-06	-3.51630e-07	0.000504113	0.000504119	1432.03	-0.159151, 0.721908
150 (cortes)	-1.25793e-06	-9.62185e-07	0.000250647	0.000250649	1994.82	-0.133167, 0.499998
150 (cortes,poi1)	-1.2654e-06	-5.43239e-07	0.00024954	0.000249544	1996.44	-0.132684, 0.498195
150 (diff1)	-7.47842e-09	2.46218e-08	4.64628e-06	4.6463e-06	382.831	-0.00527866, 0.00177874
150 (diff2)	-1.25793e-06	-2.90872e-06	0.000480375	0.000480385	244.277	-0.338330, 0.117344
175	-1.95979e-06	-1.1380e-06	0.000488034	0.000488041	1629.31	-0.165264, 0.795161
175 (cortes)	-7.17228e-07	-2.25116e-06	0.000223239	0.000223243	2239.80	-0.124684, 0.500014
175 (cortes,poi1)	-7.13530e-07	-2.6068e-06	0.000222281	0.000222285	2241.32	-0.124235, 0.498206
175 (diff1)	3.6981e-09	-1.57160e-08	4.47092e-06	4.47101e-06	396.397	-0.00535209, 0.00177226
175 (diff2)	1.24256e-06	2.86668e-06	0.000428886	0.000428893	345.81	-0.442003, 0.148314

Table 2: The statistical results of pixel brightness values for source 1 (at field center) and source 2 in FITS files of CASA-Meqtrees simulated dirty images of two 1Jy point sources separated by 75' with and without visibilities corrupted by the Cortes beam pattern model (including with the pointing error model implemented), and the difference between images with visibilities corrupted by the Cortes beam pattern model and pointing errors and visibilities corrupted with Cortes beam pattern model without pointing errors (signified as 'diff1'). Also included are the results for differenced image generated between an image with visibilities corrupted by the Cortes beam pattern model (without pointing errors) and an image with visibilities not corrupted by the Cortes beam model (signified as 'diff2'). Note that the pointing error model, signified by 'poi1', includes loffset = 0.00172179 rad = 5.919 arcmin., moffset = 0.00041211 rad = 1.416 arcmin. The statistical results for the two sources were obtained within particular small bounding box regions drawn around each source.

N_a	**mean (src1,src2)**	**median(src1,src2)**	**r.m.s. (src1,src2)**	**σ (src1,src2)**
50	1.65304e-06, 1.8582e-06	-5.37740e-06, -2.22025e-05	0.0138131, 0.0144856	0.0138133, 0.014485
50 (cortes)	1.14031e-06, -4.45598e-07	8.98684e-06, -5.57297e-06	0.006830, -4.45598e-07	0.0068309, 0.00109944
50 (cortes,poi1)	1.14255e-06, -4.50500e-07	9.59343e-06, -6.71313e-06	0.00680612, -4.50500e-07	0.00680623, 0.00107226
50 (diff1)	2.23598e-09, -4.90207e-09	-5.30417e-08, -8.7311e-10	2.83526e-05, -4.90207e-09	2.83530e-05, 0.000107113
50 (diff2)	-5.1273e-07, -2.3038e-06	-3.8038e-06, 2.56780e-05	0.00711446, -2.3038e-06	0.0071145, 0.0144052
75	3.17898e-06, 9.52520e-06	1.51374e-05, -1.59159e-05	0.00885787, 0.00837421	0.00885801, 0.00837435
75 (cortes)	3.38431e-06, -5.6606e-06	-7.12388e-06, -5.6753e-06	0.00686870, 0.00100353	0.00686881, 0.00100353
75 (cortes,poi1)	3.38661e-06, -5.67542e-06	-8.58495e-06, -5.58906e-06	0.00685851, 0.000992810	0.00685861, 0.000992811
75 (diff1)	2.30010e-09, -1.47659e-08	-9.02218e-09, 1.45751e-07	1.6070e-05, 6.35601e-05	1.60707e-05, 6.35612e-05
75 (diff2)	2.05337e-07, -1.51858e-05	8.53929e-06, -9.46596e-06	0.00352023, 0.00842339	0.00352029,0.00842352
100	2.14658e-07, 1.5098e-06	-7.73381e-06, 2.94052e-06	0.00961461, 0.0100582	0.00961476, 0.0100583
100 (cortes)	-2.08588e-08, 2.91606e-07	-8.98482e-06, 7.36631e-07	0.00479786, 0.000374081	0.00479794, 0.000374087
100 (cortes,poi1)	-1.88314e-08, 2.96061e-07	-8.02554e-06, 1.74163e-06	0.00478054, 0.000340831	0.00478062, 0.000340836
100 (diff1)	2.02732e-09, 4.45522e-09	2.33994e-08, 6.24568e-08	1.78968e-05, 7.02550e-05	1.78971e-05, 7.0256e-05
100 (diff2)	-2.3551e-07,-1.21822e-06	-2.15804e-06, -8.68266e-06	0.00483603, 0.0101065	0.00483611,0.0101066
150	-8.51284e-07, -2.03471e-08	1.1307e-05,-1.10484e-06	0.0059595, 0.00469711	0.00595961, 0.00469720
150 (cortes)	-2.01778e-07, -4.34729e-08	1.88148e-06, 5.12510e-07	0.0044159, 0.00029297	0.0044159, 0.00029298
150 (cortes,poi1)	-1.96755e-07, -4.903e-08	2.52593e-06, 9.31918e-07	0.00439999, 0.000257289	0.00440006, 0.000257294
150 (diff1)	5.02239e-09,-5.56508e-09	1.30967e-09, -2.02562e-08	1.62513e-05, 6.47303e-05	1.62516e-05,6.47314e-05
150 (diff2)	6.49506e-07, -2.31265e-08	2.71154e-06, -3.3334e-06	0.00252716, 0.00473060	0.00252720, 0.0047306
175	2.38007e-06, 1.62191e-06	-1.64189e-05, 7.2293e-06	0.00636212, 0.00546310	0.00636222, 0.00546319
175 (cortes)	1.10117e-06, -3.77661e-07	-7.56609e-06, 3.94368e-07	0.00421182, 0.000254688	0.00421189, 0.000254692
175 (cortes,poi1)	1.07323e-06, -3.88602e-07	-7.92024e-06, 8.36783e-07	0.0041966, 0.000217290	0.00419667, 0.000217294
175 (diff1)	-2.79417e-08, -1.09403e-08	9.02218e-10, 2.32919e-07	1.54359e-05, 6.03946e-05	1.54361e-05, 6.03956e-05
175 (diff2)	-1.27890e-06, -1.99957e-06	-3.29967e-06, -8.42788e-06	0.00245767, 0.00548175	0.0024577, 0.00548184

We summarize below some of the clearest and most persistent trends found among the statistical measures in each of the tables:

- For those images created from visibilities that were ***not*** corrupted by the Cortes beam model, the dynamic range estimates increased as N_a increased.
- For those images created from visibilities that ***were*** corrupted by the Cortes beam model, the dynamic range estimates also increased as N_a increased.
- For each N_a, the dynamic range estimates ***increased*** for images created from visibilities corrupted by the Cortes beam model as compared to images created from visibilities not corrupted by the Cortes beam pattern model. This may be due to the fact that root mean square

(and standard deviation about the mean) of pixel brightness values between but not including the two sources was suppressed when the Cotes beam model was implemented.

- For each N_a, the standard deviation about the mean of pixel brightness values in a bounding box region between and away from the two sources was ***smaller*** in images generated from visibilities corrupted by the Cortes beam pattern model than in images generated from visibilities ***not*** corrupted by the Cortes beam pattern model.
- For images created from visibilities not corrupted by the Cortes beam model, the standard deviation about the mean of pixel brightness values in a bounding box region between and away from the two sources generally ***decreased*** as N_a increased.
- For images created from visibilities that ***were*** corrupted by the Cortes beam model, the standard deviation about the mean of pixel brightness values in a bounding box region between and away from the two sources generally also ***decreased*** as N_a increased.
- The means and medians for pixel brightness values in the bounding box regions drawn around source 2 were significantly ***smaller*** (and also smaller compared to source 1) in images generated from visibilities corrupted by the Cortes beam model as compared to images generated from visibilities not corrupted by the Cortes beam model. This is consistent with qualitative assessments from visual examinations of the images that source 2 became significantly suppressed compared to source 1 in images generated from visibilities corrupted by the Cortes beam model.
- For all images generated from visibilities corrupted by the Cortes beam model (with and without pointing errors applied and for all N_a), the standard deviation about the mean of pixel brightness values within the bounding box region drawn around source 2 was ***smaller*** than the standard deviation about the mean of pixel brightness values within the bounding box region drawn around source 1.
- For all images generated from visibilities corrupted by the Cortes beam model (with and without pointing errors applied and for all N_a), the root mean square of pixel brightness values within the small bounding box region drawn around source 2 was ***smaller*** than the root mean square of pixel brightness values within the small bounding box region drawn around source 1.
- In at least some of the statistical results for the differenced images, the mean of pixel brightness values within the small bounding box region drawn around source 2 was ***larger*** than the mean of pixel brightness values within the bounding box region drawn around source 1. This seems to be consistent with our qualitative assessment from an examination of images that the effect of the Cortes beam model and the pointing errors (i.e., source suppression) increases further from the pointing center (e.g., there is greater suppression of source 2 than source 1).
- There was very little difference in the dynamic range of images created from visibilities corrupted by the Cortes beam model ***without*** pointing errors compared with the dynamic range of images created from visibilities corrupted with the Cortes beam model ***with*** pointing errors.

Overall, the statistical results taken together with a more qualitative evaluation of the images (e.g., visual examination) showed that source 2 was greatly suppressed for those images in which the visibilities were corrupted with the Cortes beam model (with and without pointing errors). Moreover, differenced images between images where visibilities were corrupted with the Cortes beam model and pointing errors and images with visibilities corrupted by the Cortes beam model without pointing errors show source 2 to appear brighter than source 1. The calculated imaging dynamic range values for all simulations were roughly 3 to 4 orders of magnitude smaller than dynamic range requirements for the SKA [11]. This is due, at least in part, to the fact that neither deconvolution nor corrections for primary beam effects were implemented in any of these simulations.

We carried out additional simulations using different pointing error models with different pointing offsets (errors) (e.g., l_{offset} = 0.005 rad = 17.757 arcmin, m_{offset} = 0.00123 rad = 4.250 arcmin), different sized bounding boxes regions between the sources from which statistical measures were obtained, different numbers of antennas (e.g., N_a =125 and 200), different observation and imaging parameters, and in some instances we used Meqtrees instead of CASA in generating empty measurement sets and the Meqtrees lwimager instead of the CASA imager to generate and display (FITS) image maps. The overall trends from these simulations appeared to be broadly consistent with the trends described for the above simulations. Summary tables of statistical measures similar to Table 1 above for some of these other simulations, as well as additional plots, are included in Appendix 1.

We note that in many cases the statistical measures (e.g., means, medians, etc.) were negative values because many of the pixel brightness values are negative. So, for example, in some of the difference images we are subtracting a negative pixel value from another negative pixel value, or a positive pixel value from a negative one, etc. leading to negative means and medians in the differenced images also, which can be difficult to interpret and which should be evaluated cautiously and carefully. We also note that one should use care in comparing, for the same observational and imaging simulation, statistical measures obtained from images generated when Meqtrees is used in creating the empty measurement sets and the images (with lwimager), with statistical measures obtained from images generated when CASA is used in creating the empty measurement sets and the images, as the individual statistical measures (i.e., means, medians, root mean squares, etc.) will be different even though the simulated observation is otherwise the same. These differences in statistical measures may be due to slight technical differences in the Meqtrees and CASA imagers and the implementation of imaging algorithms.

3.0 CASA A-PROJECTION ANALYSIS, TESTS, SIMULATIONS, AND IMPROVEMENTS

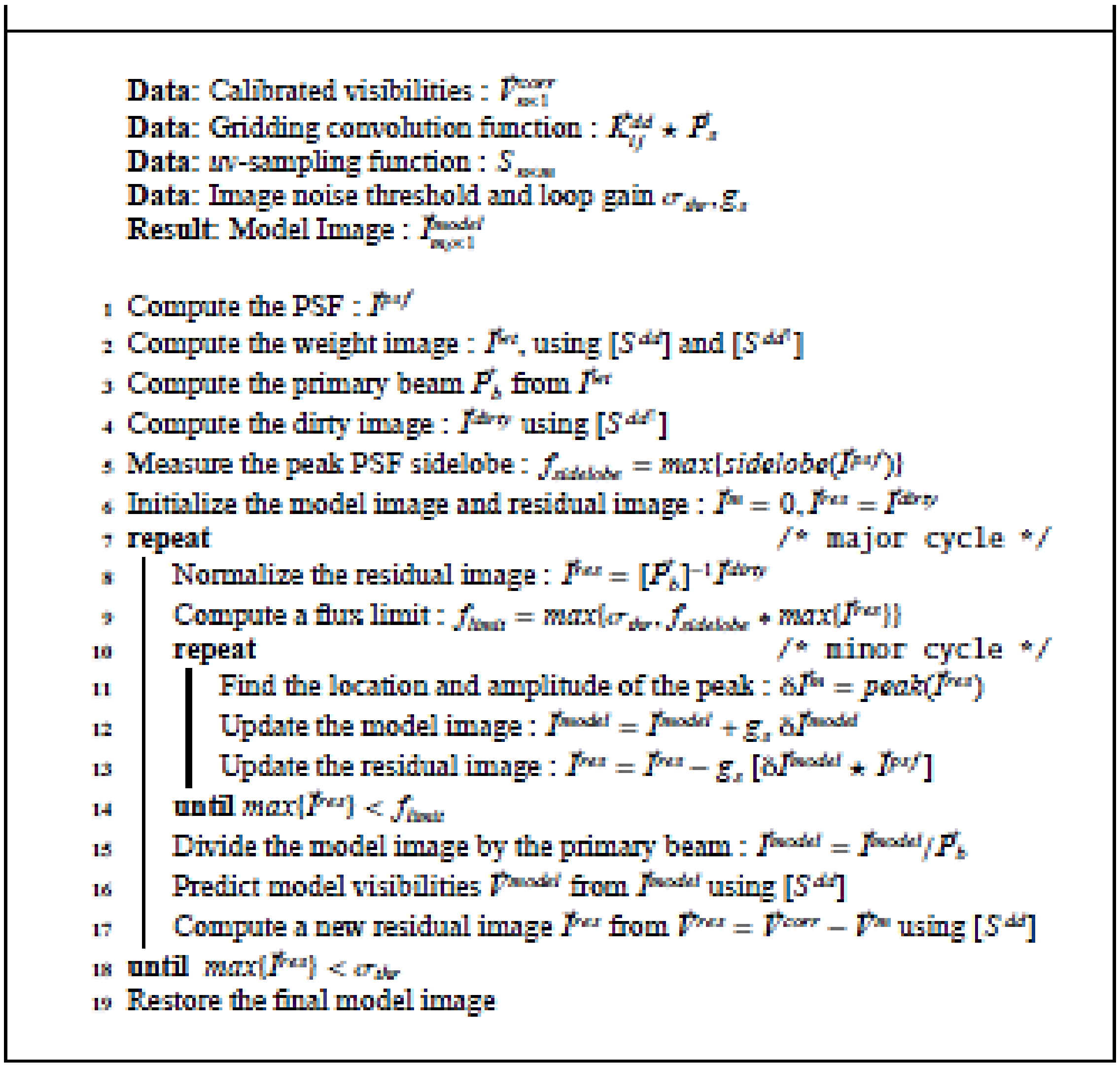
As an intermediate step toward reaching our goal of using the A-Projection algorithm to correct for the effects of the Cortes beam model in out SKA simulations, we also carried out CASA VLA simulations to better understand the implementation of A-Projection within CASA and to test and assess A-Projection's performance. We then attempted to improve the performance of A-Projection in CASA to correct for the effects of the VLA primary beam on simulated VLA observations in preparation for using CASA A-Projection to correct for primary beam effects in SKA simulations.

The A-Projection algorithm is implemented as part of the backward calculation (i.e., reverse transform) of the major cycle of the deconvolution process (e.g., Clark CLEAN). In this process, the auto-correlation of the ideal antenna illumination patterns for polarization product P are used as an interpolation operator (or, equivalently, a visibility plane filter for baseline i-j) for resampling visibilities for polarization product P on a regular grid at pixels labeled by some given indices [2,3,4]. An image corresponding to the gridded visibilities is then computed. Ultimately, the net effect of the interpolation operator is therefore averaged over all antennas for the entire range of parallactic angle coverage. The critical insight underlying A-Projection is that convolution functions can be efficiently computed both in the forward direction, during the degridding process (when predicting visibilities from a model image), or in the reverse direction, when gridding visibilities for imaging, on the condition that the convolution kernel has limited support (i.e., is significantly non-zero only within a limited region around the origin), which is the same thing as the $\mathbf{E}_p$ Jones matrix being sufficiently smooth [2].

We carried out a static and dynamic analysis of the CASA C++, Python, and Fortran code with gdb and ddd debuggers and Eclipse for C++ development to better understand the overall architecture and structure of CASA and to gain a better understanding of how the primary beam models and the A-Projection algorithm were implemented within CASA and how the A-Projection algorithm obtained the primary beam model. One can find diagrams that give a summary of the basic flow of control and function call within the CASA Imager (and related) classes for major/minor cycles and gridding/degridding (written as pseudo-code) in [12,13].

From our static analysis of CASA code, we understand that the A-Projection algorithm is implemented in the **nPBWProjectFT** C++ class. **nPBWProjectFT** does grid-based Fourier transforms which also includes the effects of primary beam and antenna pointing offsets. The **SkyEquation** class needs to be able to perform Fourier transforms on visibility data and **nPBWProjectFT** allows efficient handling of direction dependent effects due to the primary beam and antenna pointing offsets using a **VisBuffer** class which encapsulates a chunk of visibility (typically all baselines for one time) together with all the information needed for processing (e.g. UVW coordinates). The **nPBWProjectFT** class encapsulates the correction of direction dependent effects via visibility plane convolutions with a potentially different gridding convolution function for each baseline [14].

Basically, the **FTMachine** class produces the model image, and **nPBWProjectFT** is the FTMachine that is capable of dealing with primary beam corruptions and pointing offsets. **nPBWProjectFT** uses an aperture illumination function model for an antenna to calculate the convolution function used in gridding. In CASA, this gridding convolution function (GCF) includes a u-v domain convolution kernel for direction-dependent effects and a complex spheirodal function that enables the inclusion of pointing offsets as phase offsets. The convolution kernel consists of the convolution of two aperture illumination functions (AIFs) for one polarization pair [15]. The A-Projection (and AW-Projection) algorithm are described in great greater detail in [3,15,16]. Figure 12 shows pseudo-code taken from [15] of the A-Projection algorithm (similar to its implementation within CASA) as it is implemented within the CLEAN deconvolution process:



```
Data: Calibrated visibilities : $V^{corr}_{n\times 1}$
Data: Gridding convolution function : $K^{dd}_{ij} \star P_s$
Data: uv-sampling function : $S_{n\times m}$
Data: Image noise threshold and loop gain $\sigma_{thr}, g_s$
Result: Model Image : $I^{model}_{m\times 1}$

Compute the PSF : $I^{psf}$
Compute the weight image : $I^{wt}$, using $[S^{dd}]$ and $[S^{dd^T}]$
Compute the primary beam $P_b^I$ from $I^{wt}$
Compute the dirty image : $I^{dirty}$ using $[S^{dd^T}]$
Measure the peak PSF sidelobe : $f_{sidelobe} = max\{sidelobe(I^{psf})\}$
Initialize the model image and residual image : $I^m = 0, I^{res} = I^{dirty}$
repeat                                                  /* major cycle */
    Normalize the residual image : $I^{res} = [P_b^I]^{-1} I^{dirty}$
    Compute a flux limit : $f_{limit} = max\{\sigma_{thr}, f_{sidelobe} * max\{I^{res}\}\}$
    repeat                                              /* minor cycle */
        Find the location and amplitude of the peak : $\delta I^m = peak(I^{res})$
        Update the model image : $I^{model} = I^{model} + g_s\, \delta I^{model}$
        Update the residual image : $I^{res} = I^{res} - g_s\, [\delta I^{model} \star I^{psf}]$
    until $max\{I^{res}\} < f_{limit}$
    Divide the model image by the primary beam : $I^{model} = I^{model} / P_b^I$
    Predict model visibilities $V^{model}$ from $I^{model}$ using $[S^{dd}]$
    Compute a new residual image $I^{res}$ from $V^{res} = V^{corr} - V^m$ using $[S^{dd}]$
until $max\{I^{res}\} < \sigma_{thr}$
Restore the final model image
```

Figure 12: CLEAN Algorithm with Visibility Domain Corrections for Direction-Dependent Effects (A-Projection). (Taken from [15]).

We emphasize again that in the A-Projection algorithm, convolutional gridding involves the convolution of the A-Projection kernel (which is the convolution of two aperture illumination functions) with the GCF used in the standard case (e.g., the prolate spheroidal function). A-Projection convolutional resampling and convolutional gridding in CASA appears to be implemented roughly as follows[15,17,18]:

1. Forward Transform.

1.1 Initialize the 2D grid: primarily involves setup of grids and building GCFs used during gridding phase.

1.2 Initialize a cache of GCFs, oversampled by a factor of between 10 and 100 in pixel space. Recall that the GCF in A-Projection is essentially the product of 2 aperture illumination functions convolved with a prolate spheroidal wave function.

1.3 Loop over all visibility samples. (Visibility data are sampled at discrete locations in u,v space).

(a) For each visibility sample, find all points in the support region.

(b) Distribute each visibility sample value in proportion to the GCF centered on (real-valued) visibility sample locations, and evaluated at (integer-valued) grid points. In other words, scale to the correct pixel location, find the fraction of a pixel to the nearest pixel, and loop over entire support, calculating weights from the GCF.

(c) Add scaled visibilities to the grid, accumulating the sum of fractions added for later normalization.

1.4 FFT the grid, correcting for any unwanted aspects of the GCF.

2. Reverse (or transpose) Transform (FFT): Compute dirty image.

2.1 Initialize the grid.

2.2 Initialize the cache of the GCF, oversampled by a factor of between 10 and 100.

2.3 Loop over all visibility samples:

(a) For each visibility sample, find all points in the support region.

(b) Sum all grid points in the support region of each sample value in proportion to the convolution function centered on (real-valued) sample locations and evaluated at (integer-valued) grid points.

(c) Normalize by the summed weights and assign them to the visibility sample.

3.1 VLA A-Projection Simulations with CASA 3.1.0 stable binary release

As a part of our dynamic analysis of the CASA code and the implementation of A-Projection within CASA, we have carried out a number of VLA observation and imaging simulations in which simulated VLA visibilities have been corrupted by the default CASA VLA primary beam model. We have then attempted to 'correct' for these (corrupted) visibilities with the A-Projection algorithm, and then form a 'CLEANed' deconvovled image (for a 2-point-source sky model with the two sources separated by the FWHM for the VLA observation) with these 'corrected' visibilities. We have used different versions of CASA for these simulations (binary release version 3.1.0 and developer versions 3.2.0 and 3.3.0), modified parts of the CASA C++ code where A-Projection is implemented in an attempt to improve its performance, and compared the results.

For our first set of simulations, we carried out CASA VLA primary beam and A-Projection simulations on a 2-source model (with the two sources separated approximately by the FWHM for the VLA observation) with CASA stable binary release 3.1.0 to test the correct implementation and use of the default VLA PB and the correction of the corresponding corrupted visibilities by the A-Projection algorithm within CASA in advance of making modifications to the CASA C++ code to further test and improve the A-Projection algorithm in CASA. The parameters used in the VLA observation simulations included the following:

Observatory:	VLA
Field center RA:	0h7m0.0s
Field center DEC:	33d00m00s
Frequency at lower edge of band:	4300MHz
Number of Channels:	16
Channel increment:	5.0MHz
Dish diameter in meters:	25.0
Number of dishes in the array:	30
Stokes parameters:	RR RL LR LL
Integration period:	60s
Reference time:	2007/01/01
Start of observation in seconds relative to reference time:	-14400.0
Length of observation (in seconds):	28800.0
(Added) noise:	0.0Jy (no noise added)
Field of view in acrmin:	60
Number of sources to observe:	2

The VLA voltage pattern model was set up and activated in our CASA observation simulation script, and the A-Projection algorithm was set in our imaging simulation script and implemented during the imaging and CLEAN deconvolution process.

We used the Clark CLEAN and multi-field Clark CLEAN algorithm in the deconvolution process. Note that we were unable to use the Cotton-Schwab algorithm in the deconvolution process in combination with A-Projection during the imaging steps due to a possible ‘bug’ or ‘glitch’ in the CASA software, so we used the Clark CLEAN and multi-field Clark CLEAN algorithm in the deconvolution process instead. The maximum number of CLEAN iterations allowed, N_{iter}, was set to 3000, the image size was set to 576 pixels, and the pixel size was set to 0.25 arcsec.

We also increased the threshold from the default setting at which a deconvolution cycle stepped and then degrided and subtracted the model from the visibilities to form the residual, i.e., we increased the 'cyclefactor' parameter to force a major cycle more often during CLEANing. This seemed to result in A-Projection performing better, as we describe below. We also found that the A-Projection algorithm seemed to perform better (in terms of the increase in the imaging dynamic range of the resulting image map) when the multi-field Clark CLEAN algorithm was used instead of the single-field Clark CLEAN algorithm, so we used the multi-field algorithm in most of the set of simulations we describe here.

We carried out 5 imaging simulations as follows:

Simulation #1: The default VLA primary beam (PB) was turned off during the observing simulation and imaging (and so no need to correct for it); multi-field Clark CLEAN used.
Sim. #2: The default VLA PB was set and turned on during the observing simulation, but not corrected for during imaging; multi-field Clark CLEAN used.
Sim. #3: The default VLA PB was turned on and set during the observing simulation with the PB corrected for during imaging by "flux-correcting" the image (i.e., dividing the image by the "flux image"); multi-field Clark CLEAN used.
Sim. #4: The default VLA PB was turned on and set during the observing simulation and the PB corrected for by using the A-Projection algorithm during the imaging process; multi-field Clark CLEAN used.
Sim. #5: Same as Sim #4 except that the **single**-field Clark CLEAN algorithm was used during the deconvolution process.

The results of simulations #1-5 in the form of images, histogram line plots, and a summary table of image statistics are included in Figures 13-22 and Table 3 below:

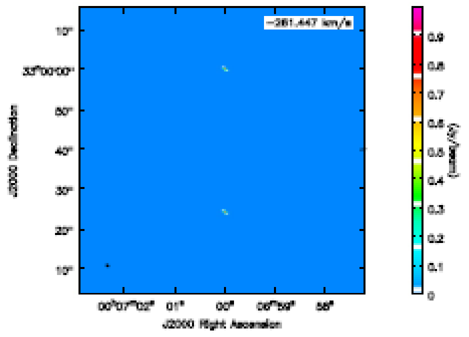
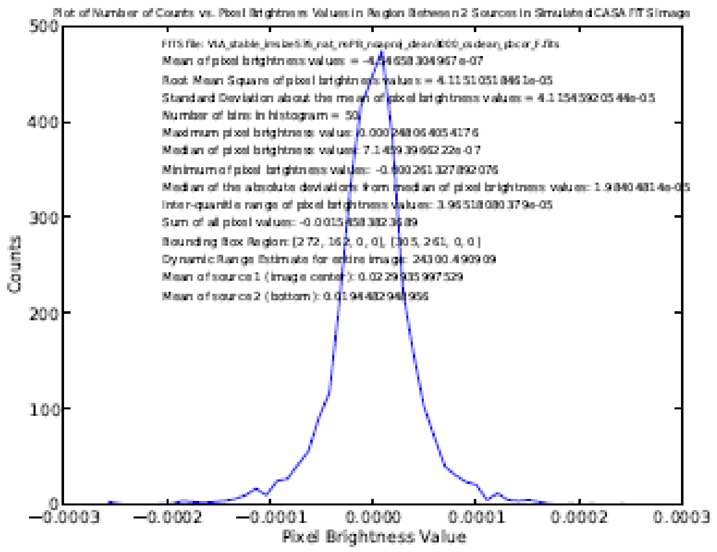


Figure 13: CASA (stable binary release 3.1.0) VLA simulation and CASA-generated

CLEANed (Niter = 3000 with Multi-field Clark CLEAN algorithm with cyclefactor = 3.0 and cyclespeedup = 50) image of two 1 Jy point sources with 0.6′ separation located at R.A.: $0^h7^m0.0^s$, Dec.:$33^d00^m00^s$ (image center) and R.A.: $0^h7^m0.0^s$, Dec.: $32^d59^m00^s$, with VLA primary beam model turned off (i.e., not used and so no need to correct for it). Simulation: Na = 30, and observing frequency of 43.0 GHz; # of freq. channels: 16; channel increment: 5.0 MHz; antenna diam.: 25.0 m.; tint = 60 sec.; Stokes parameter in image: I; imaging weights: natural; image size: 576 pixels; pixel size: 0.25 arcsec. Image displayed, and brightness-contrast colormap adjustments with casaviewer; colormap used: ’isophotes’.



```
FITS file: VLA_stable_imsize576_nat_noPB_noproj_clean3000_csclean_pbcor_F.fits
Mean of pixel brightness values = -4.54658304967e-07
Root Mean Square of pixel brightness values = 4.11510518461e-05
Standard Deviation about the mean of pixel brightness values = 4.11545920544e-05
Number of bins in histogram = 50
Maximum pixel brightness value: 0.000248064054176
Median of pixel brightness values: 7.14593966222e-07
Minimum of pixel brightness values: -0.000261327892076
Median of the absolute deviations from median of pixel brightness values: 1.98404814e-05
Inter-quantile range of pixel brightness values: 3.96518080379e-05
Sum of all pixel values: -0.00154583823689
Bounding Box Region: [272, 162, 0, 0], [305, 261, 0, 0]
Dynamic Range Estimate for entire image: 24300.490909
Mean of source 1 (image center): 0.0229935997529
Mean of source 2 (bottom): 0.0194482948956
```

Figure 14: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [272,162,0,0], top right corner = [305,261,0,0]) between but not including the two point sources in Fig. 13. Statistical measures calculated within the bounding box region are included in the plot. The dynamic range estimate is for the entire image and is taken as the ratio of the brightest (largest) positive pixel brightness value in the image to the r.m.s. of pixel brightness values in the bounding box region.

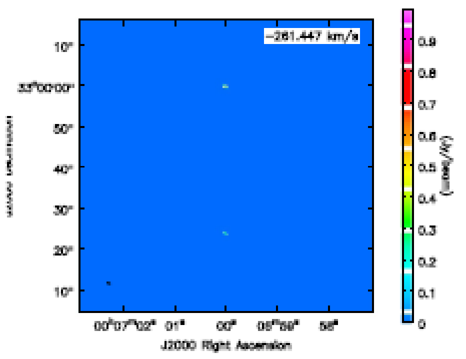
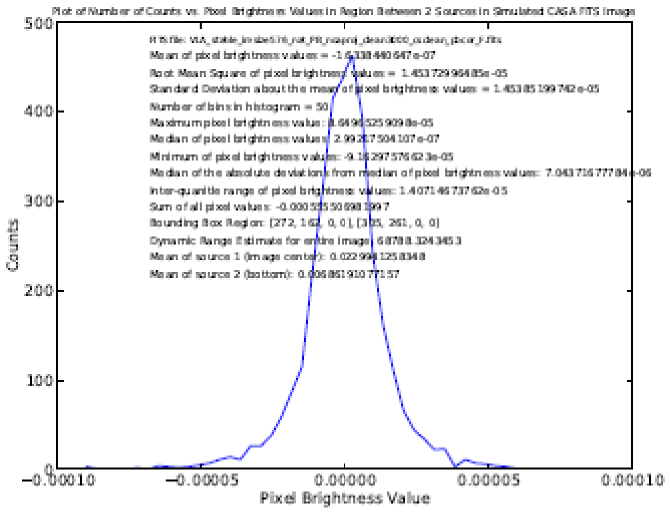


Figure 15: Same as Fig. 13 except that the VLA primary beam model has been turned on (FWHM ~1 arcmin) but not corrected for.



```
FITS file: VLA_stable_inside576_nat_PB_noapnd_clean3000_csclean_pbcor_F.fits
Mean of pixel brightness values = -1.6338440647e-07
Root Mean Square of pixel brightness values = 1.453729964 85e-05
Standard Deviation about the mean of pixel brightness values = 1.45385199742e-05
Number of bins in histogram = 50
Maximum pixel brightness value: 8.64965259098e-05
Median of pixel brightness values: 2.99217504107e-07
Minimum of pixel brightness values: -9.16297576623e-05
Median of the absolute deviations from median of pixel brightness values: 7.04371677784e-06
Inter-quartile range of pixel brightness values: 1.40714673762e-05
Sum of all pixel values: -0.000555506981997
Bounding Box Region: [272, 162, 0, 0], [305, 261, 0, 0]
Dynamic Range Estimate for entire image: 68788.3243453
Mean of source 1 (image center): 0.0229941258348
Mean of source 2 (bottom): 0.00686191077157
```

Figure 16: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [272,162,0,0], top right corner = [305,261,0,0]) between but not including the two point sources in Fig. 15.

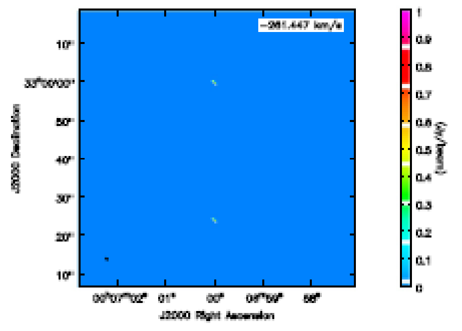
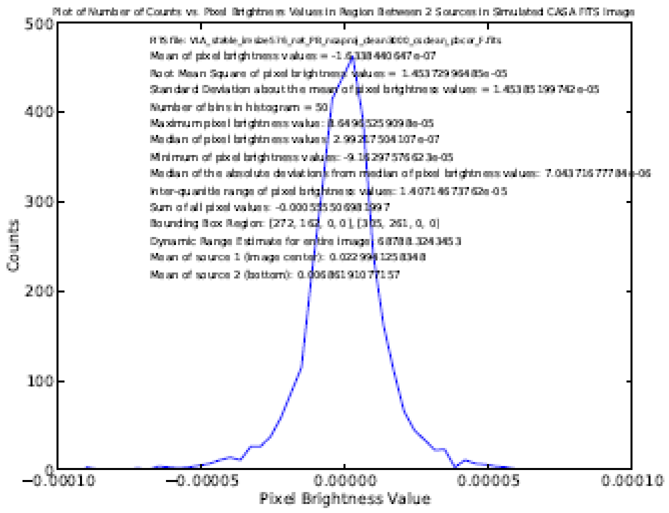


Figure 17: Same as Fig. 15 except that the VLA primary beam model has been turned on and the image has been “flux-corrected” for (i.e., divided by the “flux image”).



```
FITS file: VLA_stable_imsize576_nat_PB_noaproj_clean3000_cclean_pbcor_F.fits
Mean of pixel brightness values = -1.63338440647e-07
Root Mean Square of pixel brightness values = 1.45372996485e-05
Standard Deviation about the mean of pixel brightness values = 1.45385199742e-05
Number of bins in histogram = 50
Maximum pixel brightness value: 8.64965259098e-05
Median of pixel brightness values: 2.99217504107e-07
Minimum of pixel brightness values: -9.16297576623e-05
Median of the absolute deviations from median of pixel brightness values: 7.04371677784e-06
Inter-quarntile range of pixel brightness values: 1.40714673762e-05
Sum of all pixel values: -0.000535506981997
Bounding Box Region: [272, 162, 0, 0], [305, 261, 0, 0]
Dynamic Range Estimate for entire image: 68788.3243453
Mean of source 1 (image center): 0.0229981258348
Mean of source 2 (bottom): 0.00686191077157
```

Figure 18: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [272,162,0,0], top right corner = [305,261,0,0]) between but not including the two point sources in Fig. 17

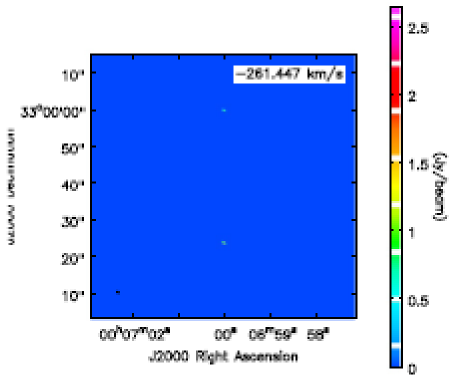
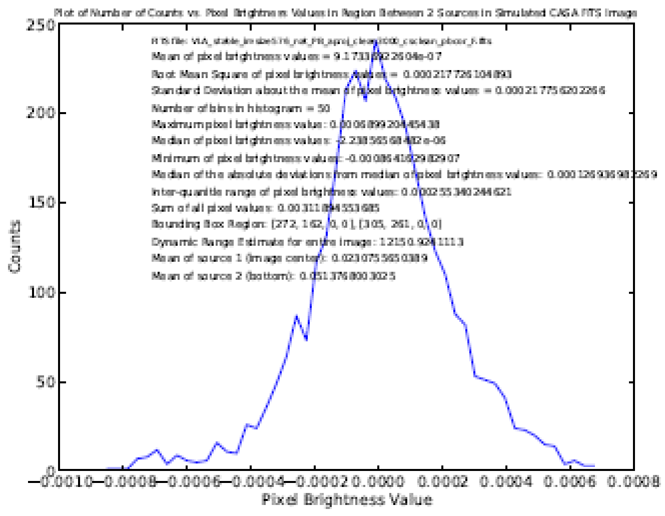


Figure 19: Same as Fig. 17 except the VLA primary beam model was turned on and corrected for with the use of the A-Projection algorithm.



```
FITS file: VLA_stable_imsize576_nat_FR_aproj_clean3000_csclean_pbcor_fits
Mean of pixel brightness values = 9.17331922604e-07
Root Mean Square of pixel brightness values = 0.000217726104893
Standard Deviation about the mean of pixel brightness values = 0.000217756202266
Number of bins in histogram = 50
Maximum pixel brightness value: 0.000689920445438
Median of pixel brightness values: -2.23856568482e-06
Minimum of pixel brightness values: -0.000864162982907
Median of the absolute deviations from median of pixel brightness values: 0.000126936982269
Inter-quartile range of pixel brightness values: 0.000255340244621
Sum of all pixel values: 0.00311894553685
Bounding Box Region: [272, 162, 0, 0], [305, 261, 0, 0]
Dynamic Range Estimate for entire image: 12150.9241113
Mean of source 1 (image center): 0.0230755650389
Mean of source 2 (bottom): 0.0513768003025
```

Figure 20: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [272,162,0,0], top right corner = [305,261,0,0]) between but not including the two point sources in Fig. 19.

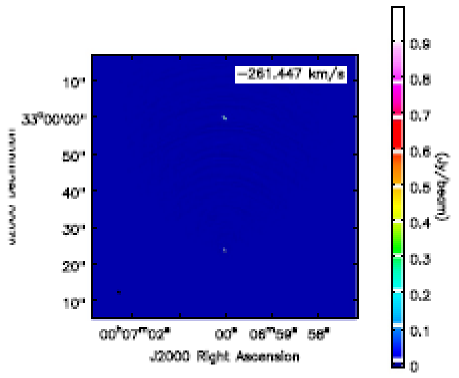
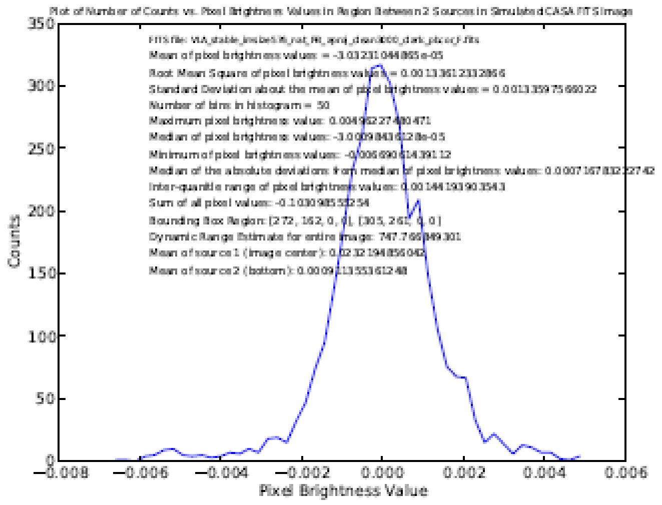


Figure 21: Same as Fig. 19 except that the single-field Clark CLEAN algorithm was used in the deconvolution process.



```
FITS file: VLA_stable_insbe5ifs_rat_fit_apnj_clean000_dark_plcor_f.fits
Mean of pixel brightness values = -3.03231044865e-05
Root Mean Square of pixel brightness values = 0.00133612332866
Standard Deviation about the mean of pixel brightness values = 0.0013359756 6022
Number of bins in histogram = 50
Maximum pixel brightness value: 0.00496227480471
Median of pixel brightness values: -3.0009843612 8e-05
Minimum of pixel brightness values: -0.00669061439112
Median of the absolute deviations from median of pixel brightness values: 0.000716783222742
Inter-quantile range of pixel brightness values: 0.0014419390354 3
Sum of all pixel values: -0.103098553254
Bounding Box Region: [272, 162, 0, 0], [305, 261, 0, 0]
Dynamic Range Estimate for entire image: 747.766849301
Mean of source 1 (image center): 0.0232194856042
Mean of source 2 (bottom): 0.00091135536124 8
```

Figure 22: Line plot of counts vs. pixel brightness value for a specified bounding box region (bottom left corner = [272,162,0,0], top right corner = [305,261,0,0]) between but not including the two point sources in Fig. 21

Table 3: Statistical results of pixel brightness values of FITS files of CASA (binary stable release version 3.1.0) simulated CLEAN images (with Niter = 3000 with use of Multi-field and Single-field Clark CLEAN with cyclefactor =3.0 and cyclespeedup = 50) of 2 1Jy point sources separated by about 0.6' located at R.A.: 0h7m0.0s, Dec.:33d00m00s (source 1, at image center) and R.A.: 0h7m0.0s, Dec.:32d59m00s (source 2), with and without VLA beam model corruption of visibilities and with and without 'flux-correction' and the A-Projection algorithm used to correct for
corrupted visibilities. The statistical results were obtained within a particular large bounding box region between the 2 sources (bottom left corner:[272,162,0,0], top right corner: [305,261,0,0]). Statistical results are also included for small bounding box regions around each of the 2 sources as well. Dynamic range estimates were obtained for the entire image. The key for the left-most column in the table is as follows: 1.Default VLA PB turned off; 2. default VLA PB turned on but not corrected for; 3. default PB turned on and image "flux-corrected" by dividing by "fluximage"; 4. default VLA PB turned on and corrected for by the A-Projection algorithm. Note that in simulations 1-4 the multi-field Clark CLEAN algorithm was used in the deconvolution process; 5. same as 4 except that the single-field Clark CLEAN algorithm was used.

PB,flux-corrected,or aproj?	mean(btwn,src1,src2)	median(btwn,src1,src2)	r.m.s.(btwn,src1,src2)	D.R.
1	-4.546e-07, 0.0229, 0.01944	7.145e-07, 5.994e-06, 2.4931e-05	4.1151e-05, 0.1080, 0.0993	24300.49
2	-1.633e-07, 0.0229, 0.0068	2.992e-07, 2.373e-06, 8.614e-06	1.453e-05, 0.1080, 0.0350	68788.32
3	-2.799e-07, 0.0229, 0.01949	3.932e-07, 2.373e-06, 2.500e-05	2.528e-05, 0.1080, 0.0995	39640.10
4	9.1733e-07, 0.0230, 0.0513	-2.2385e-06, 0.0001, 2.522e-05	0.0002, 0.1079, 0.2626	12150.92
5	-3.032e-05, 0.0232, 0.00091	-3.000e-05, 0.0006, -6.67e-05	0.0013, 0.1079, 0.0064	747.76

For Sim. #1, the mean of pixel brightness values for source 2 (src2) is slightly less than source 1 (the pixel brightness value of source 1 is 1.18 times greater than source 2), but in Sim #2 the mean brightness value for souce 2 is significantly less (by a factor of more than 3) than source #1, as we would expect. The dynamic range for Sim #2 is a little less than 3 times higher than in Sim #1. This seems somewhat counter-intuitive but may at least in part be related to the possibility that the r.m.s. in the
bounding box region between the two sources has somehow been suppressed and decreased when the un-corrected PB has been turned on and set up. The mean pixel brightness values for the two sources in Sim. #3 is about the same as in Sim. #1, and the dynamic range in Sim #3 is close to twice that in Sim #1. This seems to show that the more 'traditional' way for correcting for the effects of the PB in CASA (i.e., dividing by the "flux image") seems to be working relatively well here. In Sim #4, where A-Projection has been used to correct for the effects of the PB, the mean of pixel brightness values for source 1 is about 2.1 times greater than source 2. Also, source 2 in Sim #4 has a mean pixel brightness value that is about 1.5 times greater than in Sim #2, while the mean pixel value of source 2 (in Sim #4) is about 2.8 times less than what it is in Sims #1 and #3. Also, the dynamic
range in Sim #4 is much less (by more than a factor of 5 in one case) than in Sims #1-3.
All of this seems to be showing that correcting for the PB in CASA in the traditional way ("flux-corrected" image) still works quite a bit better than with the current implementation of A-Projection. This appears to indicate that CASA A-Projection is still under development and in need of improvement, as we would

expect the use of A-Projection to produce results (eg., in terms of imaging dynamic range) that are better than the flux-correction approach. In Sim. #5, when SINGLE-field Clark CLEAN was used in combination with A-Projection, the results seem quite poor: source 2 seems very much suppressed (it's mean pixel value is about 25 times smaller than source 1) and the dynamic range of the image is well over an order of magnitude smaller than in Sims #1-4. Thus, comparing the results of Sim. #4 with Sim. #5, we see that the use of A-Projection in combination with the multi-field Clark CLEAN algorithm produces results that are much superior to when the single-field Clark CLEAN algorithm was used in combination with A-Projection. We were not able to use the w-projection algorithm in combination with A-Projection, as this feature had not been enabled (in combination with A-Projection) in the CASA 3.1.0 binary stable release.

3.2 VLA A-Projection Simulations with CASA 3.3.0 active developer release: Modifications and improvements to the implementation of A-Projection in CASA

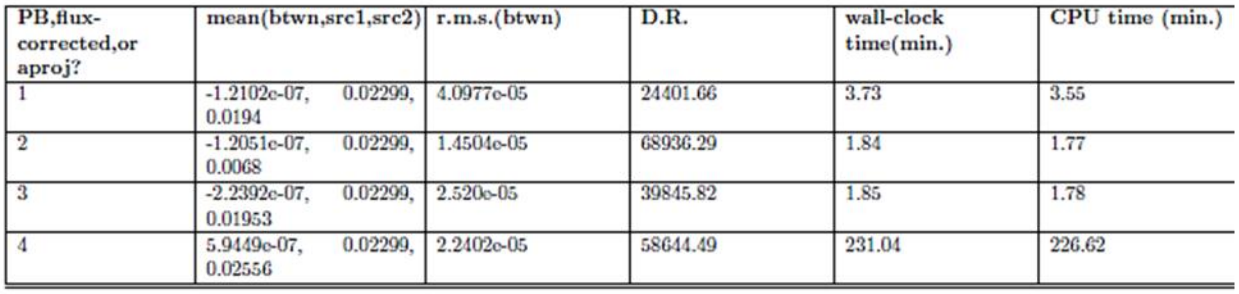
Before we can move to modify the CASA C++ code to enable A-Projection to correct for the effects of the Cortes beam model in SKA simulations, the performance of A-Projection in CASA needs to be improved. For example, we would like the dynamic range value for Sim. #4 in Table 3 (where A-Projection has been implemented) to be greater that the dynamic range value for Sim. #3 (for the "flux-corrected" image) and closer to or even greater than the dynamic range value for Sim. #2 (when the VLA primary beam was not corrected for).

It is understood that in the process of convolutional resampling and convolutional gridding in the imaging process, oversampling the GCF and increasing the support of the GCF can improve accuracy (but can increase memory access problems, the overall computational load, as well as storage) [18]. So, in our first step to improve the performance of A-Projection in CASA (e.g., in terms of higher imaging dynamic range), we (after installing CASA active developer release version 3.3.0 and un-disabling A-Projection via the front-end python tasks and tools scripts) increased the GCF oversampling factor from its default value of 20 to 32 in the CASA C++ code for the nPBWProject class (i.e., the nPBWProjectFT.cc CASA routine which implements A-Projection). We also modified and re-enabled the part of the nPBWProjectFT code (mostly by un-commenting already existing lines of code) related to the implementation of grid corrections in the utilization of the prolate spheroidal function (which is part of the GCF). We have listed these modified/re-enabled lines of code in Appendix 2.

With these modifications to the nPBWProjectFT class, we carried out the same observing and imaging CASA VLA simulations (with CASA 3.3 active developer version) as described in section 3.1 (with two point sources separated by 0.6', FWHM ~ 1', with and without the VLA PB applied and with and without PB corrections (flux corrections and A-Projection)). We have summarized the image statistics obtained from the resulting images in Table 4 below. Note that we have added two additional columns in Table 4 as compared to Tables 1-3 that contain benchmarking information: the total wall clock time and CPU time (in minutes) for the imaging simulation processing time.

Table 4: Statistical results of pixel brightness values of FITS files of CASA (active developer version 3.3.0 with the nPBWProjectFT.cc routine modified in relation to gridding correction step involving the prolate

sheroidal function and an increase in the oversampling factor for GCF to 32) simulated CLEAN images (with Niter = 3000 with use of Clark CLEAN) of 2 1Jy point sources separated by about 0.6' located at R.A.: 0h7m0:0s, Dec.:33d00m00s (source 1,at image center) and R.A.: 0h7m0:0s, Dec.:32d59m00s (source 2), with and without VLA beam model corruption of visibilities and with and without 'flux-correction' and the A-Projection algorithm used to correct for corrupted visibilities. Statistical results obtained within particular large bounding box region between the 2 sources (bottom left corner:[272,162,0,0], top right corner: [305,261,0,0]). Statistical results are also included for small bounding box regions around each of the 2 sources as well. Dynamic range estimates were obtained for entire image. CASA VLA simulation: Na = 30; observing freq.: 43.0 GHz; # of freq. channels: 16; channel increment: 5.0 MHz; antenna diam.:25.0 m.; tint = 60 sec.; FWHM of PB: ~ 1.0 arcmin; Stokes parameter in image: I; imaging weights: natural; image size: 576 pixels; pixel size: 0.25 arcsec. The key for the left-most column in the table is as follows: 1.Default VLA PB turned on; 2. default VLA PB turned on but not corrected for; 3. default PB turned on and image “flux-corrected" by dividing by “flux image"; 4. default VLA PB turned on and corrected for by A-Projection algorithm.



PB,flux-corrected,or aproj?	mean(btwn,src1,src2)	r.m.s.(btwn)	D.R.	wall-clock time(min.)	CPU time (min.)
1	-1.2102e-07, 0.02299, 0.0194	4.0977e-05	24401.66	3.73	3.55
2	-1.2051e-07, 0.02299, 0.0068	1.4504e-05	68936.29	1.84	1.77
3	-2.2392e-07, 0.02299, 0.01953	2.520e-05	39845.82	1.85	1.78
4	5.9449e-07, 0.02299, 0.02556	2.2402e-05	58644.49	231.04	226.62

We can see from Table 4 that the modifications in the nPBWProjectFT code have resulted in significant improvements (i.e., increases) in the dynamic range of the resulting simulated images when the A-Projection algorithm was used to correct for the effects of the VLA PB (i.e., the dynamic range for Sim. #4 in Table 4 is almost 5 times greater that the dynamic range for Sim. #4 in Table 3), along with even less suppression of source 2. We can also see that the mean pixel count brightness for source 2 in this same simulated image is 0.02556 (Sim. #4), which is now actually greater than the mean pixel brightness of source 1 (at field center).

We also note that these modifications of the CASA code have essentially no effect on the results of imaging simulations when the A-Projection algorithm is not used, as can be seen when we compare Sim. #1-3 in Table 4 with Sim #1-3 in Table 3. Thus, the modifications to the the nPBWProject FT class discussed earlier (see also Appendix 3.2) seem to be directly tied in with the implementation of A-Projection. We found that increasing the oversampling factor had only a very minor effect on the results (in terms of increase in the dynamic range and suppression of source #2). The reason why these lines of code in nPBWProjectFT.cc may have been commented out in the first place by CASA developers is that they seem to lead to a significant increase in the processing time (due at least in part to a significant increase in the number of CLEAN major cycles that are carried out during the deconvolution process) for images to be generated when A-Projection was utilized, as can be seen in the 'wall-clock time' and 'CPU time' columns of Table 4: the wall clock and CPU time in Sim. #4 was more than two orders of magnitude greater that in Sim. #2 and 3.

In our next set of CASA VLA A-Projection simulations we modified additional lines of code in nPBWProjectFT.cc related to the prolate spheroidal wave function gridding correction steps. In this case, we focused on lines of code related to grid corrections in anticipation of convolution by the GCF. Specificly, we focused on a section of the code where gridded visibility values are being divided by a primary beam 'correction factor', i.e.,

```
 Int iy=lix.position()(1);
   gridder->correctX1D(correction,iy);

   Vector<Float> PBCorrection(liavgpb.rwVectorCursor().shape()),
     avgPBVec(liavgpb.rwVectorCursor().shape());

   PBCorrection = liavgpb.rwVectorCursor();
   avgPBVec = liavgpb.rwVectorCursor();
   for(int i=0;i<PBCorrection.shape();i++)
     {
       //
       // This with the PS functions
       //
             //
        PBCorrection(i)=FUNC(avgPBVec(i))*sincConv(i)*sincConv(iy);
        if ((abs(PBCorrection(i))) >= pbLimit_p)
                   //lix.rwVectorCursor()(i) /= PBCorrection(i);
                   // lix.rwVectorCursor()(i) /= sqrt(sqrt(PBCorrection(i)));
                   lix.rwVectorCursor()(i) /= pow(PBCorrection(i),1.0/2.05);
                 //lix.rwVectorCursor()(i) /= sqrt(PBCorrection(i));
```

and

```
PBCorrection(ix) = FUNC(PBCorrection(ix))/(sincConv(ix)*sincConv(iy));
   if ((abs(PBCorrection(ix))) >= pbLimit_p)
              // {lix.rwVectorCursor()(ix) /= sqrt(PBCorrection(ix));}
             {lix.rwVectorCursor()(ix) /= pow(PBCorrection(ix),1.0/2.05);}
```

where 'griddedVis = lix.rwVectorCursor()' and 'PBCorrection=lipb.rwVectorCursor()'. We found that the dynamic range and the amount by which source 2 is suppressed by the VLA PB in simulations is very sensitive, at least within certain ranges of values, to the power to which the value of PBCorrection(i) or PBCorrection(ix) function is taken in the denominator in the equations above when A-Projection is implemented. So, for example, the achievable dynamic range seems to peak when taking PBCorrection(...) to the power of 1/2.0 or 1/2.05 but then seems to decline for 1/2.1, 1/2.2, 1/3.0, etc. We carried out simulations where PBCorrection(...) has been taken to the power of 1.0, 1/1.5, 1/1.75, 1/2.0, 1/2.05, 1/2.1, 1/2.2, and 1/3.0, e.g.,

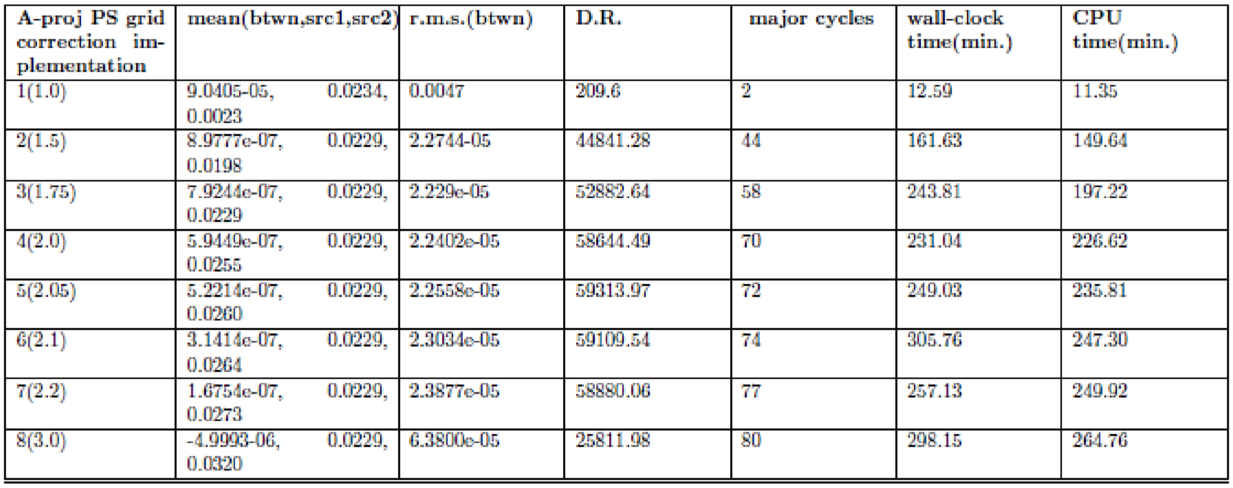
```
 {lix.rwVectorCursor()(ix) /= pow(PBCorrection(ix),1.0/2.05);}
```

and we've summarized the results of the image statistics in Table 5, which includes a detailed caption that describes the key in the left-most column of the table. The highest dynamic range of a simulated image we've been able to obtain so far is when PBCorrection(...) is set to the power of 2.05, i.e.,

```
{lix.rwVectorCursor()(ix) /= pow(PBCorrection(ix),1.0/2.05);}
```

and the dynamic range was 59313.97 (~6.0e4). Table 5 also includes a column for the number of CLEAN major cycles that were carried out (as was automatically determined within CASA) in each simulation.

Table 5: Statistical results of pixel brightness values of FITS file images from CASA 3.3.0 VLA observation and imaging simulations (with same parameters and settings as in the simulations for Table 3) for different implementations of A-Projection to correct for visibilities corrupted by the VLA PB. The different implementations of A-Projection were set by modifying lines of code in nPBWProjectFT.cc related to gridding correction steps involving the prolate spheroidal (PS) funcion and PS function gridding correction steps in anticipation of convolution by the GCF (and where each polarization plane is corrected by the appropriate PB). The key for the left-most column of the table is as follows: 1. Essentially no modifications made to nPBWProjectFT.cc for PS function gridding correction steps (i.e., gridded visibilities divided by PB correction factor to the power of 1.0). 2. Modifications to PS function gridding correction steps including gridded visibilities divided by the PB correction factor to the power of 1.5. 3. Same as 2 except gridded visibilities divided by the PB correction factor to power of 1/1.75. 4. Same as 3 except gridded visibilities divided by the PB correction factor to power of 1/2.0. 5. Same as 4 except gridded visibilities divided by the PB correction factor to the power of 1/2.05. 6. Same as 5 except gridded visibilities divided by the PB correction factor to power of 1/2.1. 7. Same as 6 except gridded visibilities divided by the PB correction factor to power of 1/2.2. 8. Same as 7 except gridded visibilities divided by PB correction factor to power of 1/3.0.



A-proj PS grid correction implementation	mean(btwn,src1,src2)	r.m.s.(btwn)	D.R.	major cycles	wall-clock time(min.)	CPU time(min.)
1(1.0)	9.0405-05, 0.0234, 0.0023	0.0047	209.6	2	12.59	11.35
2(1.5)	8.9777e-07, 0.0229, 0.0198	2.2744-05	44841.28	44	161.63	149.64
3(1.75)	7.9244e-07, 0.0229, 0.0229	2.229e-05	52882.64	58	243.81	197.22
4(2.0)	5.9449e-07, 0.0229, 0.0255	2.2402e-05	58644.49	70	231.04	226.62
5(2.05)	5.2214e-07, 0.0229, 0.0260	2.2558e-05	59313.97	72	249.03	235.81
6(2.1)	3.1414e-07, 0.0229, 0.0264	2.3034e-05	59109.54	74	305.76	247.30
7(2.2)	1.6754e-07, 0.0229, 0.0273	2.3877e-05	58880.06	77	257.13	249.92
8(3.0)	-4.9993-06, 0.0229, 0.0320	6.3800e-05	25811.98	80	298.15	264.76

The overall trend of the data summarized in Table 5 seems to be that as the exponent of PBCorrection(...) decreases from 1.0 to 1/2.05 or so, the dynamic range of the resulting simulated image increases (as do the processing times and number of major cycles, for the most part) and reaches it's peak. As the exponent of PBCorrection (...) further decreases beyond 1/2.05, the dynamic range of the resulting simulated images decreases (though the number of CLEAN major cycles and processing times seem to continue to increase to some extent).

In our final set of CASA (version 3.3.0) VLA PB simulations in which A-Projection was used to correct for the visibilities corrupted by the VLA PB, we used the A-Projection algorithm in combination with the w-projection algorithm. We had found that the ability to use A-Projection in combination with w-projection was not enabled in CASA versions 3.1 and 3.2. However, we found that we were able to use A-Projection in combination with w-projection with CASA version 3.3.0 active developer release without any error message or segmentation fault issues. The observation and imaging parameters for each simulation was the same as in Table 5, as were the modifications to the nPBWProjectFT class. In the case of this set of simualations, the prolate spheroidal function gridding correction step modifications included dividing the gridded visibilities by the primary beam correction factor to the power of 2.049 in each of the simulations. We also increased the size of the GCF oversampling factor by a very small amount as compared to the previous set of simulations (Table 5). The results, given in Table 6 below, are again in terms of the image statistics measures of each image, in which A-Projection was used together with w-projection for each simulation. The left-most column of the table shows the number of w-projection planes used in each simulation.

Table 6: Statistical results of pixel brightness values of FITS files images from CASA 3.3.0 VLA observation and imaging simulations (with the same parameters and settings as in Table 4) for implementations of A-Projection together with w-projection to correct for visibilities corrupted by the VLA PB. The implementation of A-Projection was set by modifying lines of code in nPBWProjectFT.cc related to gridding correction steps involving the prolate spheroidal (PS) funcion and PS function gridding correction steps in anticipation of convolution by the GCF (and where each polarization plane is corrected by the appropriate PB). Specifically, the PS function gridding correction step modifications included dividing the gridded visibilities by the PB correction factor to the power of 2.049 in all cases. The leftmost column of the table lists the number of W-Projection planes for each simulation. Note that the oversampling factor for GCF was set to 32.8 in all cases.

no. w-projection planes	mean(btwn,src1,src2)	r.m.s.(btwn)	D.R.	no. major cycles	wall-clock time(hrs.)	CPU time(hrs.)
2	8.9959e-07, 0.0229, 0.0259	2.5693e-5	5.1936e4	67	4.66	4.39
4	3.2522e-07, 0.0229, 0.0260	2.0516e-05	6.5213e4	73	5.47	4.97
5	3.2298e-07, 0.0229, 0.0260	2.0460e-05	6.5389e4	73	6.05	5.22
6	3.0292e-07, 0.0229, 0.0260	2.0442e-05	6.5447e4	73	5.42	5.20
8	3.3139e-07, 0.0229, 0.0259	2.1718e-05	6.1541e4	72	6.93	5.49
128	-3.6085e-07, 0.0230, 0.0256	3.44547e-05	3.8487e4	65	27.65	22.74

The statistical data show that when a small number of W-Projection planes are used (e..g, 5-7) there is a slight increase in dynamic range of the resulting simulated images on top of the already significant increase in dynamic range which occurred after applying the particular implementations of A-Projection as described earlier involving modifications to the nPBWProjectFT class. However, the dynamic range begins to decrease again when the number of W-Projection planes increases above 6-7, with a somewhat more significant decrease in dynamic range by the time we are up to 128 w-projection planes. Note that this is in contrast toTable I of [10], where w-projection is used alone (and not in combination with A-Projection) in wide-field VLA imaging simulations (where primary beam effects are not taken into account). For the case of those simulations, dynamic range values increase steadily from 8 to 128 w-projection planes. For out simulations, on the other hand, with A-Projection used together with w-projection, there appears to be a decrease in dynamic range with increasing number of W-Projection planes (at least between 8 and 128 w-Projection planes). The difference between our results and [10] deserves further analysis and study.

We can also see from Table 6 that there is a very significant increase in the wall-clock and CPU times with 128 W-projection planes as compared to when 6 or 8 W-Projection planes are used. Note that the dynamic range value for when 6 W-Projection planes were used (~6.5e4) in Table 6 is nearly as large as the dynamic range for Sim. #2 of Table 3 (6.8e4) when the default VLA PB was used in the simulation without being corrected for by A-Projection, **and** the mean pixel brightness value of source 2 in Table 6 is almost 4 times greater (much less suppressed) than in Sim. #2 in Table 3. The improvements in the performance of A-Projection that we have achieved can also be seen by comparing the results of Table 6 with the results of Sim. #3 and 4 in Table 3. However, the dynamic range values in Table 6 are still somewhat less than dynamic range values known to be achievable by the VLA in some instances [11], so there is still room for improvement in the implementation of A-Projection in CASA.

4.0 FUTURE WORK

This document has described Meqtrees and CASA SKA and VLA observation and imaging simulations that are important steps toward our final goal of carrying out SKA simulations with CASA in which the visibilities are corrupted by the Cortes Beam model and Fourier plane corrections are then implemented by the A-projection algorithm to reverse or correct for the primary beam effects. Work like this is important in assessing the performance of A-Projection (or other next-generation imaging algorithms that correct for the effects of antenna primary beams) against different antenna and interferometer design parameters, as well as different computing hardware and software architectures that will need to handle unprecedented data flows and volumes in both wide-field imaging and time-domain radio astronomy. Such work is necessary in the development and construction of next generation radio synthesis arrays like SKA that require increased algorithmic performance in terms of sensitivity and imaging dynamic range over existing telescopes, as well as the development of associated computing hardware and software systems that can handle the data challenges [4,5].

4.1 Importing and/or Adding the Cortes Primary Beam Model to CASA

Many common voltage pattern (VP) and primary beam models have been coded into CASA. Currently, recognized models include the VLA, ATCA, GBT, GMRT, WRST, and Hat Creek. However, the Cortes beam model (or any other VP/PB model for the SKA) does not currently exist within the CASA library of VP/PB models. Therefore, we must be able to import and/or add in some way the Cortes beam model to CASA in order to meet the final goals of our project. There are at least three primary ways that we can do this:

(1) Carry out an SKA observation simulation with CASA without any primary beam corruptions. Then, take the resulting CASA-generated empty measurement set and use it within Meqtrees to corrupt the visibilities with the Cortes beam model. Then, use these corrupted visibilities to create an image map with CASA. This allows CASA to generate images from the corrupted visibilities. In this sense, visibilities corrupted by the Cortes beam model have been 'brought into' CASA (even though the Cortes beam model itself has technically not been added to the CASA primary beam model library).
(2) Use a number of already-existing CASA front-end tools and tasks, mostly in the form of python scripts and XML files, to, for example, make a VP/PB from a user-supplied image or a user-supplied vector, or to save a VP/PB description as a table.
(3) Modify already existing CASA C++ classes or write new C++ classes to add and implement parameters for the SKA telescope, SKA observations, and the Cortes beam model within CASA.

We have successfully carried out option #1 using Tony Willis' python code and python TDL scripts, as described in Section 2.0. However, option #1 involves the additional complexity of using and configuring two separate software packages. The use of two different software packages may also lead to greater computational loads and processing times than using the CASA software package alone, especially when very large SKA measurement sets are involved. Furthermore, option #1 alone would not enable the use of A-Projection to correct visibilities for the effects of the Cortes beam model, due in part to the fact that A-Projection has not yet been implemented within Meqtrees.

We have investigated and experimented with option #2 as well. According to the CASA documentation (http://casa.nrao.edu/), there are front-end tasks and tools that one can use to make a VP/PB from a user-supplied image or a user-supplied vector without necessarily modifying any CASA classes (C++ code). For example, we have generated FITS file images of the Cortes beam model from Tony Willis' scripts, and we attempted to read in these images and generate Cortes PB data from them within CASA (using such tasks as 'importfits' and 'setpbimage') for use with the SKA observing simulations, but
we have received error messages such as the following when trying to do this:

```
2011-12-08 19:22:16     SEVERE  PBMath2DImage::checkJonesCongruent      Primary beam images must
have 4 polarization values: number is 1
2011-12-08 19:22:16     SEVERE  Simulator::createSkyEquation() (file
/home/yashar/casa/active/code/synthesis/implement/MeasurementEquations/Simulator.cc, line 2186)
Caught exception: 2011-12-08 19:22:16    SEVERE  PBMath2DImage::checkJonesCongruent      Primary
beam images must have 4 polarization values: number is 1
2011-12-08 19:22:16     SEVERE  Simulator::predict() (file
/home/yashar/casa/active/code/synthesis/implement/MeasurementEquations/Simulator.cc, line
2106)     Failed to create SkyEquation
```

These errors seem to indicate, at least in part, that the FITS file images of the Cortes beam model generated from Tony Willis' python code have a structure and format which is different from and/or incompatible with that needed by CASA, so we may also need to find a way to modify Willis' python code to generate Cortes beam image FITS files that are compatible with CASA. It is also not clear to us at this time if these or similar front-end tasks are fully functional within CASA in general or for our specific purposes. For example, the 'setpbimage' task, which makes a VP/PB from a user-supplied image, is apparently still in an 'experimental' stage of implementation. In any case, this task along with other tasks such as 'setpbnumeric', 'importfits', 'createantresp', and 'saveastable' may be useful for the purposes of reaching the goals of this project. If these tasks are fully functional and can be successfully implemented, then they could allow us to:

- create a VP/PB model for use within CASA from a user supplied image (setpbimage).
- create a VP/PB model for use within CASA from a user supplied vector (setpbnumeric).
- create and save VP or PB descriptions as a beam pattern table (saveastable). This table can then be used to implement primary beam corrections when creating images.
- import a FITS image into CASA image table format (importfits).

It may also be necessary to modify these tasks via their CASA python scripts and XML files or even write new tasks.

Option #2 alone may not be adequate for implementing A-Projection to correct for the effects of the Cortes beam model. It may be necessary to pursue option #3 and modify CASA C++ classes such as nPBWProjectFT and VLACalcIlluminationConvFunc to be successful in reaching our goals. Through the use and implementation of the BeamCalc class, VLACalcIlluminationConvFunc makes use of a feed pattern text file containing three columns of angle (in degrees) and power (in dBi) VLA data. This data is used in the VLA aperture illumination function (AIF) computations needed for use with the gridding convolution function (GCF). The nPBWProjectFT class then uses this VLA AIF model to calculate the convolution function used in gridding, which, in turn, is used in the implementation of A-Projection. Given that the Fourier transform of the AIF gives the voltage pattern, we believe that it may be possible to modify the VLACalcIlluminationConvFunc class or write a new similar class that can read from an image file of the Cortes beam model and then use the resulting image data to generate aperture illumination function model data for the SKA Cortes beam model. The nPBWProjectFT class could then use this SKA (Cortes beam) AIF model to calculate the GCF for use in the implementation of A-Projection. Given that such work may require significant modifications to the CASA C++ code, it may be advantageous for the SKA project team to enter into a collaboration with the CASA team similar to the ALMA-CASA collaboration so that the work and expertise of CASA developers can be drawn upon to write the necessary CASA C++ classes for the implementation of A-Projection to correct for the effects of the Cortes beam model in SKA simulated observations.

4.2 Alternatives to A-Projection

A-Projection provides an accurate method to apply known direction-dependent effects when predicting visibilities from a model image, and an approximate method for correcting them when imaging [2,3,16]. However, the limitations of A-Projection are poorly understood. In particular, it is not clear how or whether dynamic range is limited by (a) non-unitarity of the $\mathbf{E}_p$ Jones matrices, and (b) the fact that high-order terms in the convolution kernels (which are Fourier transforms of the Sky-Jones terms $\mathbf{E}_p$) are ignored (i.e., the limited support assumption) [2]. Therefore, it may also be useful to explore and experiment with different algorithms and techniques for correcting for direction-dependent effects such as primary beam effects. Alternatives and possible supplements to the A-Projection algorithm for correcting for primary beam effects include:

- The differential gain approach [2, 19].
- Subtraction in the u-v plane or the Discrete Fourier Transform approach; also referred to as image-plane corrections [2,4].
- The peeling algorithm [2,20].
- Pointing self-calibration (Pointing SelfCal algorithm) [4,21].

- Facet imaging. [2,22].

The differential gain approach is closely connected to peeling. It might be thought of as a generalized, simultaneous form of peeling [2]. They are also very similar in that they attempt to solve for the same phenomenological effect: a direction-dependent gain term [2]. The differential gains approach has been demonstrated using Meqtrees via an example of extremely high dynamic range calibration of WSRT observations of 3C 147 at 21cm, with full treatment of direction-dependent effects [2,19]. A version of the radio interferometry measurement equation incorporating differential gains has been implemented in Meqtrees and is publicly available [19]. The major drawback of peeling is that it can be very expensive computationally [2]

DFT approaches [2,4] have the advantage of maximum precision (at least to the extent that the direction-dependent effects are known) but are very computationally expensive since they scale linearly with the number of sources being modeled. A-Projection is approximate, but its computational cost scales much better, since it only depends on resolution [2]. The DFT and A-Projection approaches may be complementary and can be favorably combined (provided compatible implementations are available) by using DFTs for the brighter sources in the field and A-Projection for the fainter sources. By choosing a flux threshold, one could then achieve a clear trade-off between accuracy (and, thus, dynamic range) and computational cost [2].

The Pointing SelfCal algorithm iteratively solves for antenna pointing errors in the visibility domain, where it is efficient to compute the gridding convolution operator parameterized by pointing errors [3,21]. Pointing SelfCal is implemented in an experimental version of CASA [2].

The major drawback of facet imaging is the high computing cost when many facets are involved, and the fact that time variability in $\mathbf{E}_p$ cannot be taken into account [2].

4.3 Computing Costs and Challenges

In [23] we summarized the computational costs in forming dirty images of a number of existing radio interferometry imaging algorithms used to deal with non-coplanar baselines in wide-field radio interferometry, including:

- Fourier Sums
- 3-D Fast Fourier Transform (FFT)
- Facets/Polyhedron imaging
- W-Projection
- Hybrid facets and W-Projection

In the future, we would like to supplement and expand upon this work by considering the computational costs of w-stacking imaging [18], the A-Projection algorithm and other algorithms used to correct for primary beam effects as discussed in Section 4.3, deconvolution operations, and parallel computing efficiency. In all cases the effective computational cost is a strong function of the efficiency with which the algorithm can be mapped to the underlying computing architecture, including the computational intensity, memory footprint, and memory bandwidth requirements, as well as other factors such as gridding operations and the size of gridding convolution functions [17,23]. For example, when examining the entries under wall clock time and CPU time in Tables 5 and 6, one can imagine the advantages of repeating the simulations discussed in this paper with parallelized CASA C++ code on Linux clusters or parallel computers and supercomputers such as the Blue Waters system at the University of Illinois, Urbana-Champaign.

4.4 Radio Imaging Statistics Project

As a complement to the work described and suggested in this document, we would like to carry out a radio imaging statistics analysis project at NCSA or elsewhere in which Portable Batch System batch job scripts are to be submitted to Linux cluster systems, and which will include the following tasks:

(1) generating a simulated radio image separately with each computing node,
(2) corrupting each image with gain error models described in [1],
(3) distributing Monte Carlo or bootstrap samples across unique hosts,
(4) computing statistics for each simulated image, and
(5) computing final statistics for all images.

The eventual goal of the project will be to calibrate the relation between how these gain errors (i.e., amplitude, phase, pointing, and beam-width errors) translate into r.m.s. errors, and to verify corresponding analytical dynamic range expressions derived in [1]. This work will involve batch job scheduling on Linux clusters and utilizing and implementing (parallelized) C++ MPI code.

APPENDICES

Appendix 1 – Supplementary Material for Section 2.0: Additional Plots and Summary Tables

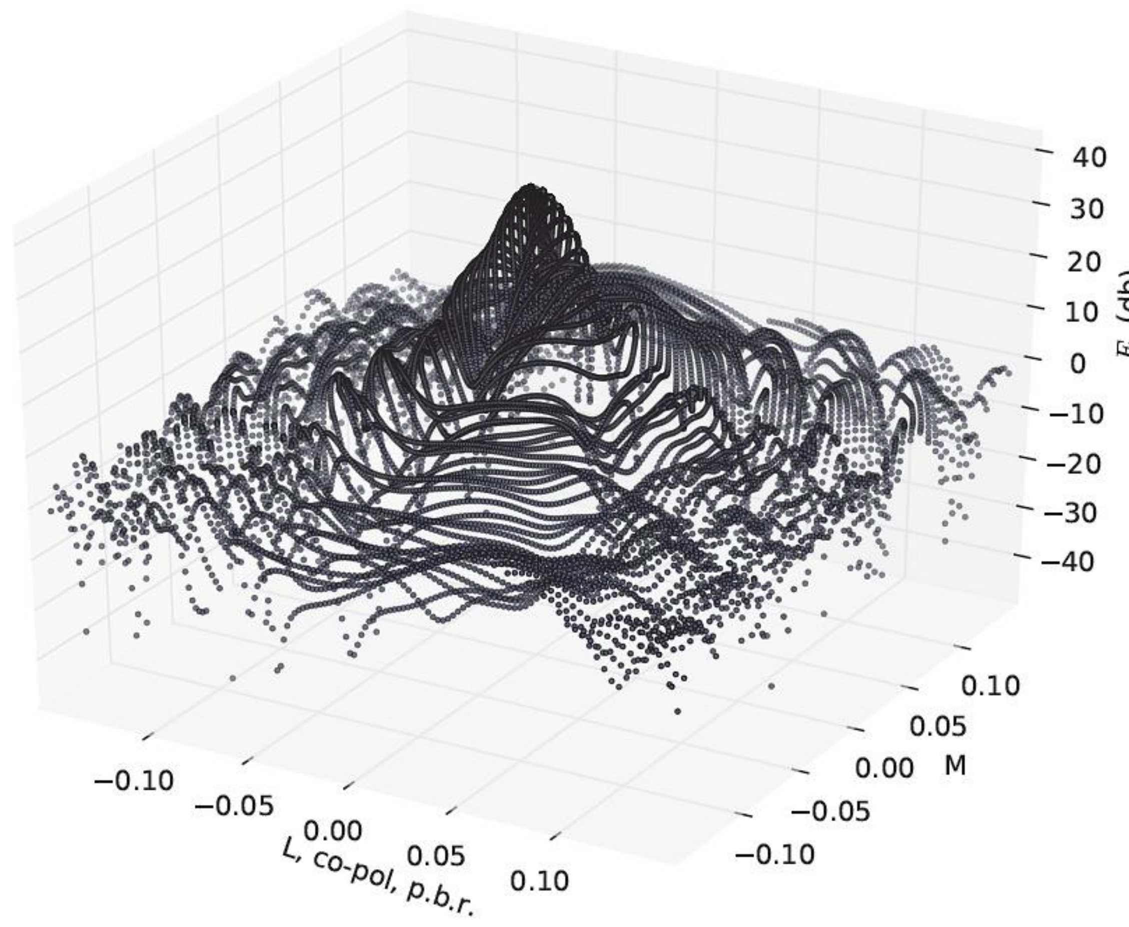


Figure 1.1: 3D scatter plot of Cortes primary beam response (E_θ; co-polarization; in units of db) as a function of (L,M) coordinates (radians) on the sky. Corresponding to Figure 2, with subreflector illumination angle of 67 degrees.

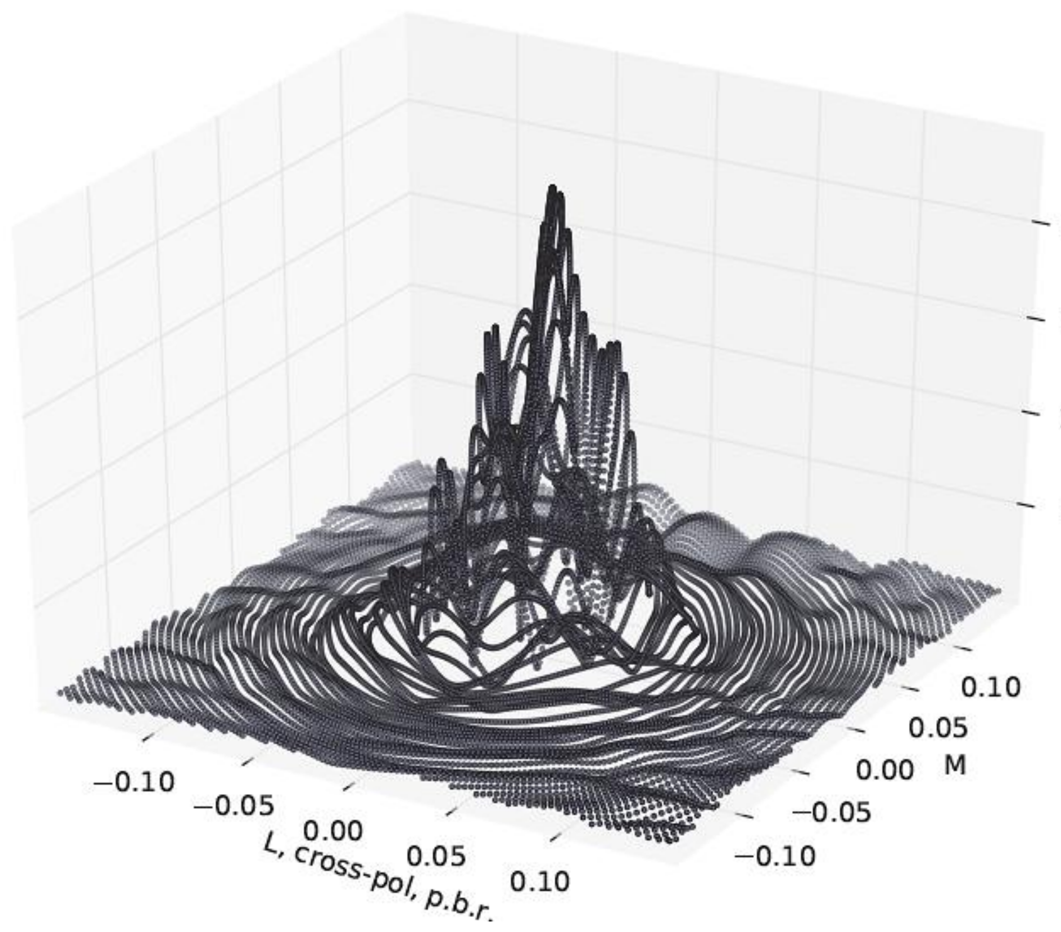


Figure 1.2: 3D scatter plot of Cortes primary beam response (E_ϕ; cross-polarization; in units of db) as a function of (L,M) coordinates (radians) on the sky. Corresponding to Figure 3, with subreflector illumination angle of 42 degrees.

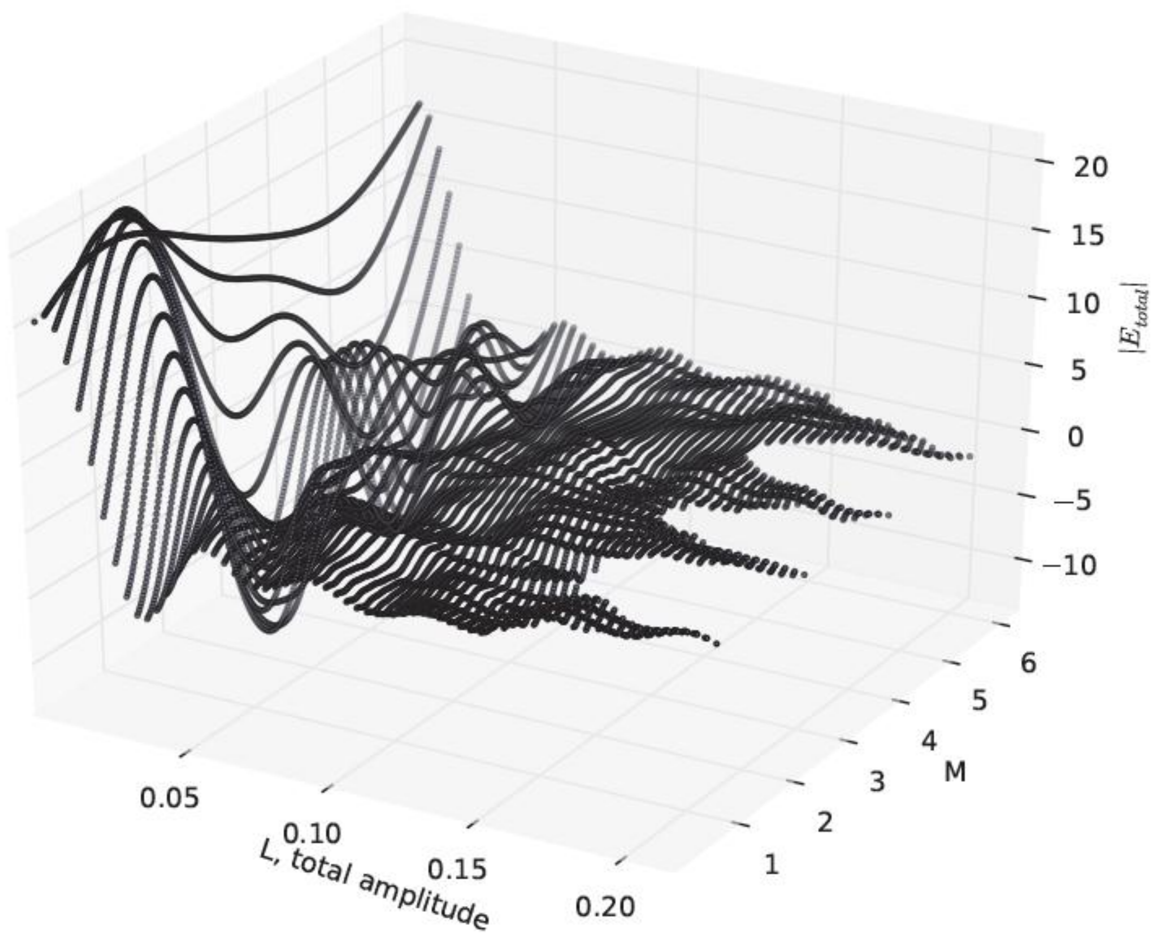
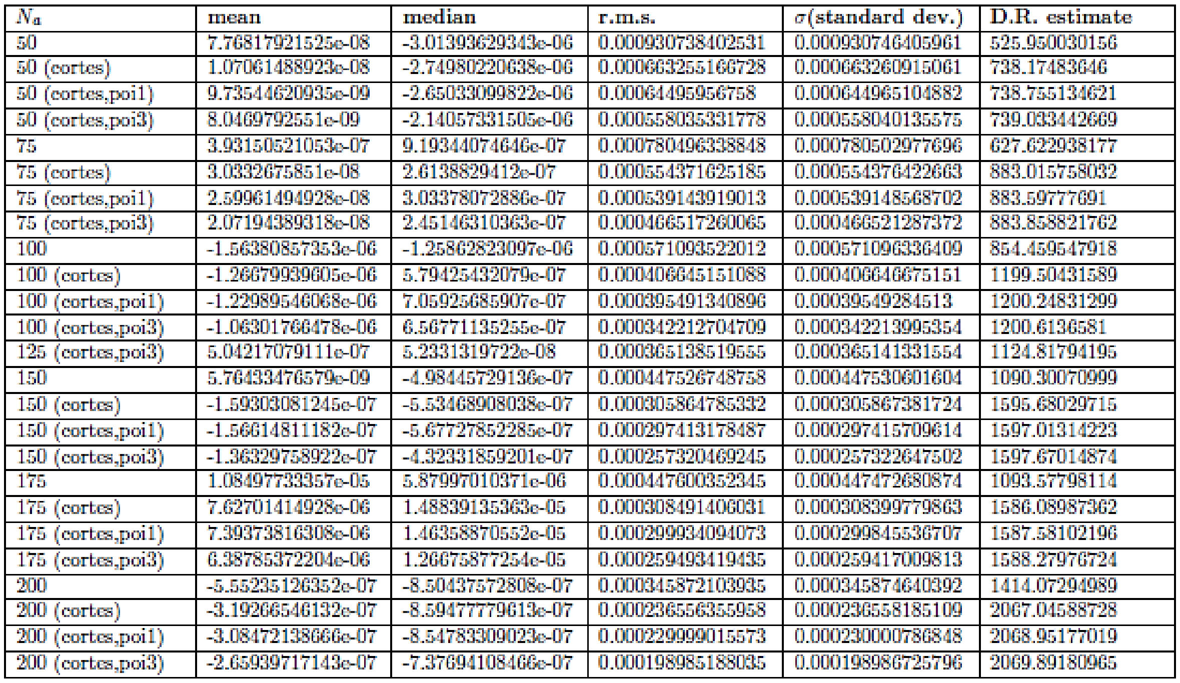


Figure 1.3: 3D scatter plot of total magnitude of Cortes beam power pattern (in units of db) as a function of (L,M) coordinates (radians) on the sky. Corresponding to Figure 3, with subreflector illumination angle of 42 degrees.

Table 1.1: Statistical results of pixel brightness values of FITS files of Meqtrees simulated dirty images of 2 1Jy point sources separated by 75’ located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s,with and without Cortes beam pattern applied (including with pointing error model applied). Pointing error models: poi1=(l_{offset} = 0.00172179 rad = 5.919 arcmin, m_{offset} = 0.00041211 rad = 1.416 arcmin), poi3=(l_{offset} = 0.005 rad = 17.757 arcmin, m_{offset} = 0.00123 rad = 4.250 arcmin). Statistical results obtained within particular large bounding box region between the 2 sources (bottom left corner:[977,514,0,0], top right corner: [1102,973,0,0]); dynamic range estimate was obtained for entire image. Meqtrees SKA simulations: log-spiral configs. with Na = 50, 75, 100, 150, 175, 200; field center: R.A.=0h0m0.1s, Dec.=28d0m1s; observing freq.: 1400 MHz; # of freq. channels: 64; channel increment: 5.0 MHz; antenna diam.: 12.0 m.; tint = 60 sec.; scan length: 90 min.; FWHM of primary beam: ~75’; Stokes parameter in image: I; imaging weights: uniform; number of convolution functions for w-proj.: 128; image size: 2048 pixels, 273 arcmin. Note: CASA was not used at all in any of these simulations.



N_a	mean	median	r.m.s.	σ(standard dev.)	D.R. estimate
50	7.76817921525e-08	-3.01393629343e-06	0.000930738402531	0.000930746405961	525.950030156
50 (cortes)	1.07061488923e-08	-2.74980220638e-06	0.000663255166728	0.000663260915061	738.17483646
50 (cortes,poi1)	9.73544620935e-09	-2.65033099822e-06	0.00064495956758	0.000644965104882	738.755134621
50 (cortes,poi3)	8.0469792551e-09	-2.14057331505e-06	0.000558035331778	0.000558040135575	739.033442669
75	3.93150521053e-07	9.19344074646e-07	0.000780496338848	0.000780502977696	627.622938177
75 (cortes)	3.0332675851e-08	2.6138829412e-07	0.000554371625185	0.000554376422663	883.015758032
75 (cortes,poi1)	2.59961494928e-08	3.03378072886e-07	0.000539143919013	0.000539148568702	883.59777691
75 (cortes,poi3)	2.07194389318e-08	2.45146310363e-07	0.000466517260065	0.000466521287372	883.858821762
100	-1.56380857353e-06	-1.25862823097e-06	0.000571093522012	0.000571096336409	854.459547918
100 (cortes)	-1.26679939605e-06	5.79425432079e-07	0.000406645151088	0.000406646675151	1199.50431589
100 (cortes,poi1)	-1.22989546068e-06	7.05925685907e-07	0.000395491340896	0.00039549284513	1200.24831299
100 (cortes,poi3)	-1.06301766478e-06	6.56771135255e-07	0.000342212704709	0.000342213995354	1200.6136581
125 (cortes,poi3)	5.04217079111e-07	5.2331319722e-08	0.000365138519555	0.000365141331554	1124.81794195
150	5.76433476579e-09	-4.98445729136e-07	0.000447526748758	0.000447530601604	1090.30070999
150 (cortes)	-1.59303081245e-07	-5.53468908038e-07	0.000305864785332	0.000305867381724	1595.68029715
150 (cortes,poi1)	-1.56614811182e-07	-5.67727852285e-07	0.000297413178487	0.000297415709614	1597.01314223
150 (cortes,poi3)	-1.36329758922e-07	-4.32331859201e-07	0.000257320469245	0.000257322647502	1597.67014874
175	1.08497733357e-05	5.87997010371e-06	0.000447600352345	0.000447472680874	1093.57798114
175 (cortes)	7.62701414928e-06	1.48839135363e-05	0.000308491406031	0.000308399779863	1586.08987362
175 (cortes,poi1)	7.39373816308e-06	1.46358870552e-05	0.000299934094073	0.000299845536707	1587.58102196
175 (cortes,poi3)	6.38785372204e-06	1.26675877254e-05	0.000259493419435	0.000259417009813	1588.27976724
200	-5.55235126352e-07	-8.50437572808e-07	0.000345872103935	0.000345874640392	1414.07294989
200 (cortes)	-3.19266546132e-07	-8.59477779613e-07	0.000236556355958	0.000236558185109	2067.04588728
200 (cortes,poi1)	-3.08472138666e-07	-8.54783309023e-07	0.000229999015573	0.000230000786848	2068.95177019
200 (cortes,poi3)	-2.65939717143e-07	-7.37694108466e-07	0.000198985188035	0.000198986725796	2069.89180965

Table 1.2: Same as Table 1.1 except that statistical results were obtained within a particular smaller bounding box region between the 2 sources (bottom left corner:[982,732,0,0], top right corner: [1084,841,0,0]).

N_a	**mean**	**median**	**r.m.s.**	**σ(standard dev.)**	**D.R. estimate**
50	2.16230343677e-07	1.61857940384e-06	0.000619171361905	0.000619198640302	790.608094943
50 (cortes)	3.56144440504e-07	-2.66793904302e-06	0.000480101763969	0.00048012281278	1019.78020281
50 (cortes,poi1)	3.48110603814e-07	-2.43598242378e-06	0.000466812198283	0.000466832662178	1020.68282261
50 (cortes,poi3)	3.02240419421e-07	-1.52195104874e-06	0.000403866375564	0.000403884084964	1021.14659038
75	7.30412880527e-07	2.68822941507e-06	0.00053043582011	0.000530458702169	923.499859648
75 (cortes)	3.2508630776e-07	-5.87100487337e-07	0.000393933587475	0.000393950847253	1242.6431673
75 (cortes,poi1)	3.12299523207e-07	-1.01987518519e-06	0.000383200269425	0.00038321704445	1243.17858385
75 (cortes,poi3)	2.68120592233e-07	-7.08226366442e-07	0.000331617309712	0.000331631848841	1243.40733652
100	-2.98288962308e-06	-3.19067567034e-06	0.000376957556	0.000376962383797	1294.51261785
100 (cortes)	-3.71460884954e-06	-3.94830840378e-06	0.000284970185021	0.000284958553621	1711.66191905
100 (cortes,poi1)	-3.6218410829e-06	-3.34754668074e-06	0.000277205457678	0.000277194035163	1712.40428918
100 (cortes,poi3)	-3.13809921392e-06	-2.65007020062e-06	0.000239884248003	0.000239874307314	1712.76459655
125 (cortes,poi3)	1.25311873176e-06	1.30847411128e-06	0.00026227926719	0.000262287858789	1565.94290693
150	-2.50858178612e-07	4.98085114486e-07	0.000280693056993	0.000280705327629	1738.33559382
150 (cortes)	-7.84830229459e-08	-8.76920410064e-07	0.000206533426535	0.000206542525601	2363.11583909
150 (cortes,poi1)	-7.51671322381e-08	-1.08813901534e-06	0.000200863607461	0.00020087245991	2364.65311323
150 (cortes,poi3)	-6.49028925389e-08	-9.60250645221e-07	0.000173798893229	0.000173806551552	2365.45368463
175	-3.75046695007e-05	-3.63220278814e-05	0.000301659369143	0.000299332054651	1622.64441202
175 (cortes)	-5.95697310834e-06	-3.86303236155e-06	0.00022476814047	0.000224699104347	2176.88812206
175 (cortes,poi1)	-5.51818248915e-06	-4.25445568908e-06	0.000218563698581	0.000218503669307	2178.63112072
175 (cortes,poi3)	-4.66388640484e-06	-3.75178456125e-06	0.000189106911421	0.000189057736003	2179.44518645
200	-2.10922075583e-06	-1.62517051194e-06	0.000218565779505	0.00021856524987	2237.71711841
200 (cortes)	6.65402409352e-08	-5.04722038386e-07	0.000163130709552	0.000163137888725	2997.42975455
200 (cortes,poi1)	8.42158490909e-08	-6.31659531791e-07	0.000158618044225	0.00015862502323	3000.01725995
200 (cortes,poi3)	8.1018165276e-08	-5.52561857603e-07	0.000137230788823	0.00013723682047	3001.35133294

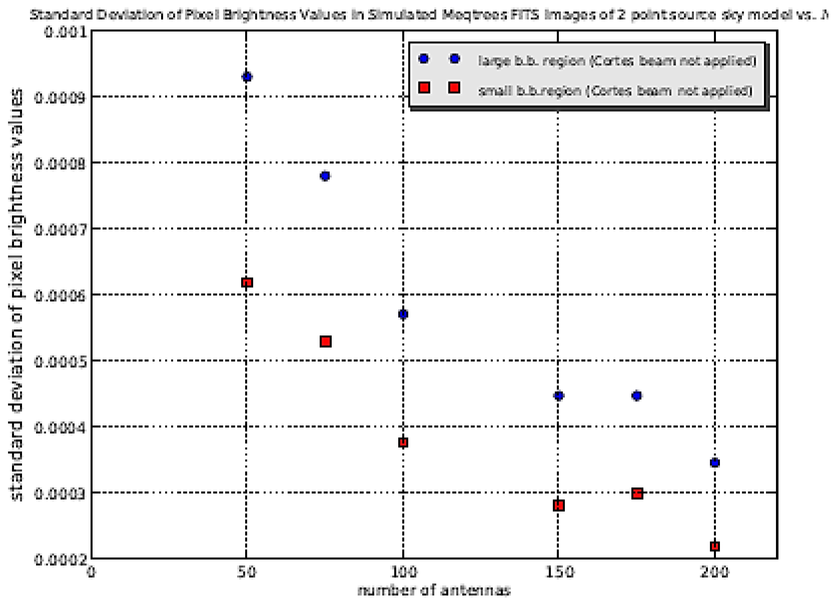


Figure 1.4: Plot of standard deviation about the mean of pixel brightness values vs. number of antennas for two bounding box regions for Meqtrees SKA simulations and Meqtrees generated dirty images of two 1 Jy point sources with 75' separation, located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s, without Cortes beam applied. Simulation: log-spiral configuration; field center: R.A.=0h0m0.1s, Dec.=28d0m1s; observing frequency: 1400 MHz; number of frequency channels: 64; channel increment: 5.0 MHz; antenna diameter: 12.0 m.; integration time: 60 sec.; scan length: 90 min.; noise: 0.0 Jy; FWHM of primary beam: ~ 75'; Stokes parameter in image: I; imaging weights: uniform; number of convolution functions for w-projection: 128; image size in pixels: 2048; image size in arcmin: 273.

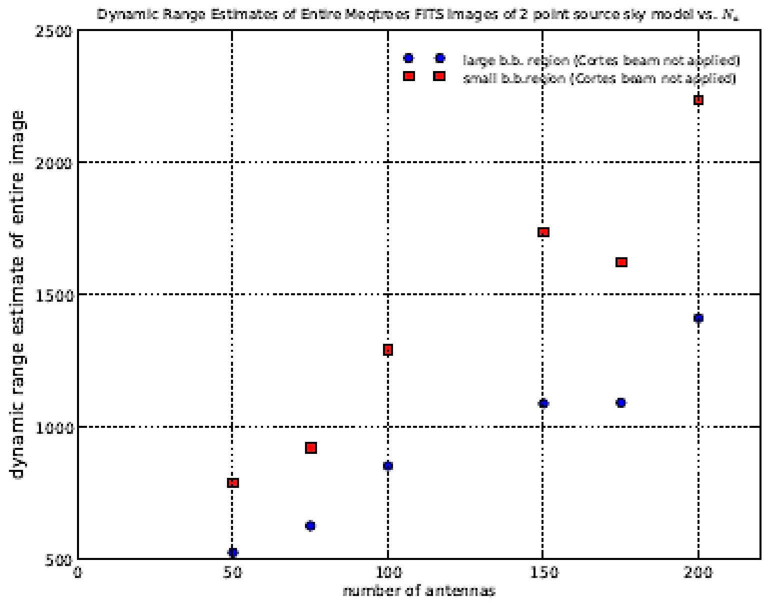


Figure 1.5: Same as Fig. 1.4 except that plot is for dynamic range estimate vs. number of antennas. Dynamic range is calculated here as ratio of peak brightness in all of image to r.m.s. of pixel brightness values in respective bounding box regions.

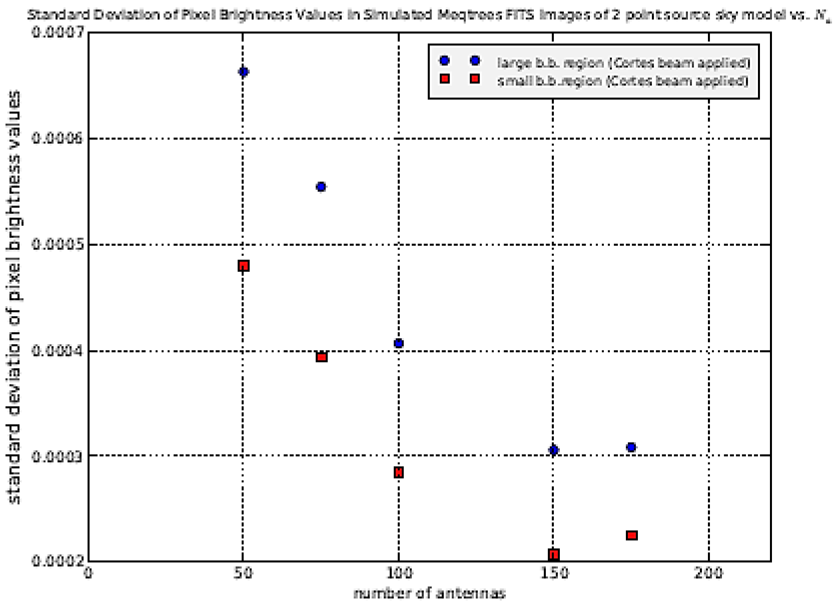


Figure 1.6: Same as Fig. 1.4 except that Cortes primary beam pattern model has been applied in the simulations .

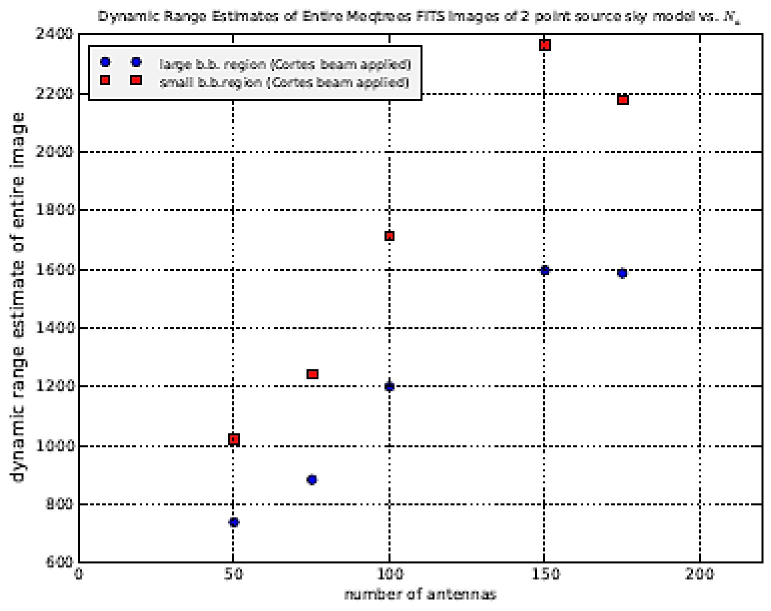


Figure 1.7: Same as Fig. 1.6 except that plot is for Dynmaic Range estimate vs. Number of Antennas.

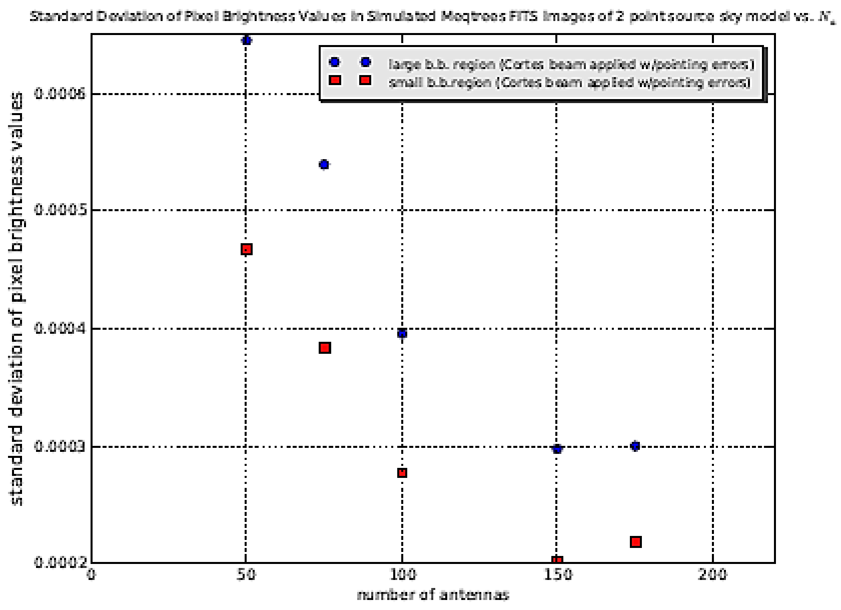


Figure 1.8: Same as Fig. 1.6 except that the pointing error model has also been applied with the

Cortes beam primary beam pattern model.

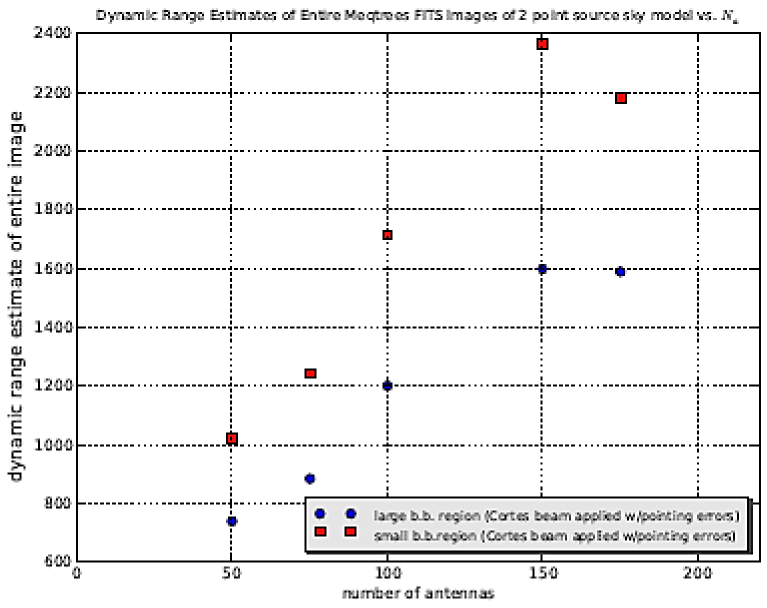


Figure 1.9: Same as Fig. 1.8 except plot is for Dynamic Range estimate vs. Number of Antennas.

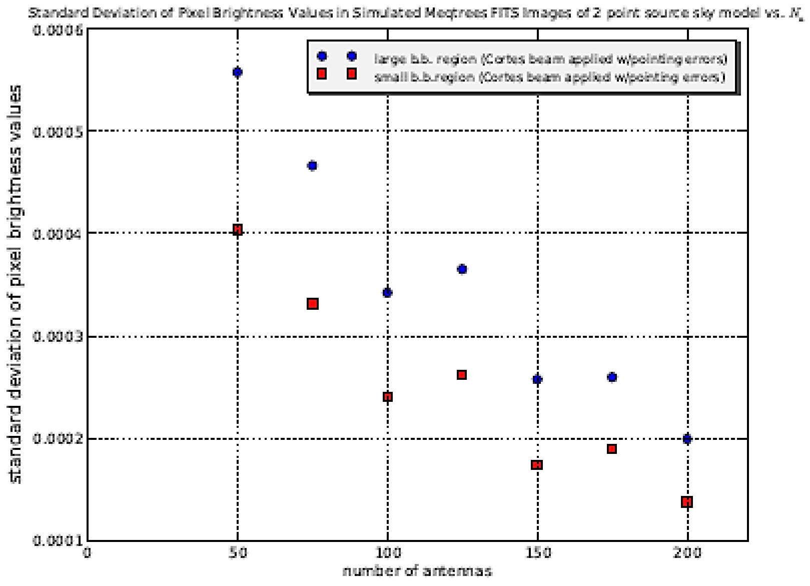


Figure 1.10: Plot of standard deviation about the mean of pixel brightness values vs. number of antennas for two bounding box regions for Meqtrees SKA simulations and Meqtrees-generated dirty images of two 1 Jy point sources with 75' separation, located at R.A.: 0h0m0.1s, Dec.:28d0m1s and R.A.: 0h0m0.1s, Dec.:26d45m10s, with Cortes beam and scaled-up AGW pointing error model (l_{offset} = 0.005 rad = 17.7570, m_{offset} = 0.00123 rad = 4.2500) applied. Simulation: log-spiral configuration; field center: R.A.=0h0m0.1s, Dec.=28d0m1s; observing frequency: 1400 MHz; number of frequency channels: 64; channel increment: 5.0 MHz; antenna diameter: 12.0 m.; integration time: 60 sec.; scan length: 90 min.; noise: 0.0 Jy; FWHM of primary beam: ~ 75 arcmin; Stokes parameter in image: I; imaging weights: uniform; number of convolution functions for w-projection: 128; image size in pixels: 2048; image size in arcmin: 273.

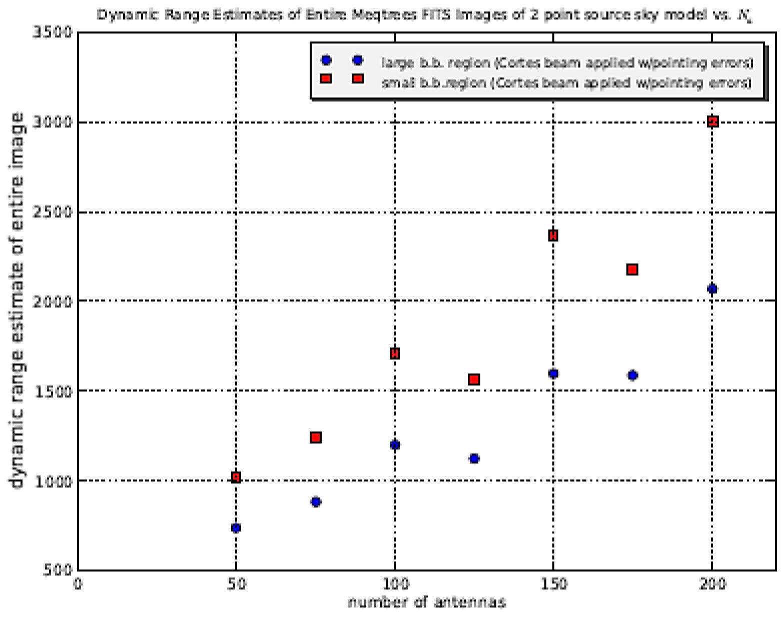


Figure 1.11: Same as Fig. 1.10 except that plot is for dynamic range estimate vs. number of antennas. Dynamic range is calculated here as ratio of peak brightness in all of image to r.m.s. of pixel brightness values in respective bounding box regions.

Appendix 2 – Supplementary Material for Section 3.2

We have listed below modified, re-enabled, and related code from the CASA nPBWProjectFT class connected to the implementation of grid corrections in the utilization of the prolate spheroidal (PS) wave function as discussed in Section 3.2. Note that the prolate spheroidal wave function is a part of the GCF for the implementation of A-Projection.

```
// /*
     // screen=0.0;
     // First the spheroidal function
     //
     // Int inner=convSize/2;
     //        screen = 0.0;
     Vector<Complex> correction(inner);
     for (Int iy=-inner/2;iy<inner/2;iy++)
       {
         ggridder.correctX1D(correction, iy+inner/2);
         for (Int ix=-inner/2;ix<inner/2;ix++)
           screen(ix+convSize/2,iy+convSize/2)=correction(ix+inner/2);
         if (iy==0)
            for (Int ii=0;ii<inner;ii++)
```

```
                cout << ii << " " << correction(ii) << endl;
      }
    // */

.....


    // Grid correct in anticipation of the convolution by the
    // convFunc.  Each polarization plane is corrected by the
    // appropraite primary beam.
    //
    for(lix.reset(),lipb.reset();!lix.atEnd();lix++,lipb++)
      {
        Int iy=lix.position()(1);
        gridder->correctX1D(correction,iy);
        griddedVis = lix.rwVectorCursor();

        Vector<Float> PBCorrection(lipb.rwVectorCursor().shape());
        PBCorrection = lipb.rwVectorCursor();
        for(int ix=0;ix<nx;ix++)
          {
            // PBCorrection(ix) = (FUNC(PBCorrection(ix)))/(sincConv(ix)*sincConv(iy));

            //
            // This is with PS functions included
            //
            //if (doPBCorrection)
            //{
            //  PBCorrection(ix) = FUNC(PBCorrection(ix))/(sincConv(ix)*sincConv(iy));
            //  //PBCorrection(ix) = FUNC(PBCorrection(ix))*(sincConv(ix)*sincConv(iy));
            //  if ((abs(PBCorrection(ix)*correction(ix))) >= pbLimit_p)
            //    {lix.rwVectorCursor()(ix) /= (PBCorrection(ix))*correction(ix);}
            //  else
            //    {lix.rwVectorCursor()(ix) *= (sincConv(ix)*sincConv(iy));}
            //}
            //else
            //lix.rwVectorCursor()(ix) /= (correction(ix)/(sincConv(ix)*sincConv(iy)));
            //
            // This without the PS functions

            if (doPBCorrection)
             {
                   PBCorrection(ix) = FUNC(PBCorrection(ix))/(sincConv(ix)*sincConv(iy));
              //PBCorrection(ix) = FUNC(PBCorrection(ix))*(sincConv(ix)*sincConv(iy));
              //  PBCorrection(ix) = (PBCorrection(ix))*(sincConv(ix)*sincConv(iy));
              //  lix.rwVectorCursor()(ix) /= (PBCorrection(ix));
              if ((abs(PBCorrection(ix))) >= pbLimit_p)
                {lix.rwVectorCursor()(ix) /= sqrt(PBCorrection(ix));}
                  //{lix.rwVectorCursor()(ix) /= (PBCorrection(ix));}
              else
                {lix.rwVectorCursor()(ix) *= (sincConv(ix)*sincConv(iy));}
            }
             else
            lix.rwVectorCursor()(ix) /= (1.0/(sincConv(ix)*sincConv(iy)));

          }
        }



....

// Apply the gridding correction
        //
        {
          normalizeAvgPB();
          Int inx = lattice->shape()(0);
          Int iny = lattice->shape()(1);
          Vector<Complex> correction(inx);

          Vector<Float> sincConv(nx);
          Float centerX=nx/2;
          for (Int ix=0;ix<nx;ix++)
            {
              Float x=C::pi*Float(ix-centerX)/(Float(nx)*Float(convSampling));
              if(ix==centerX) sincConv(ix)=1.0;
              else            sincConv(ix)=sin(x)/x;
            }

          IPosition cursorShape(4, inx, 1, 1, 1);
          IPosition axisPath(4, 0, 1, 2, 3);
          LatticeStepper lsx(lattice->shape(), cursorShape, axisPath);
          LatticeIterator<Complex> lix(*lattice, lsx);
          LatticeStepper lavgpb(avgPB.shape(),cursorShape,axisPath);
          //LatticeStepper lavgpb(sensitivityImage.shape(),cursorShape,axisPath);
          LatticeIterator<Float> liavgpb(avgPB, lavgpb);
          //LatticeIterator<Float> liavgpb(sensitivityImage, lavgpb);
          for(lix.reset(),liavgpb.reset();
              !lix.atEnd();
```

```
  lix++,liavgpb++)
{
  Int pol=lix.position()(2);
  Int chan=lix.position()(3);

  if(weights(pol, chan)>0.0)
    {
      Int iy=lix.position()(1);
      gridder->correctX1D(correction,iy);

      Vector<Float> PBCorrection(liavgpb.rwVectorCursor().shape()),
        avgPBVec(liavgpb.rwVectorCursor().shape());

      PBCorrection = liavgpb.rwVectorCursor();
      avgPBVec = liavgpb.rwVectorCursor();

      for(int i=0;i<PBCorrection.shape();i++)
        {
          //
          // This with the PS functions
          //
          //PBCorrection(i)=FUNC(avgPBVec(i))*sincConv(i)*sincConv(iy);
          //if ((abs(PBCorrection(i)*correction(i))) >= pbLimit_p)
          //lix.rwVectorCursor()(i) /= PBCorrection(i)*correction(i);
          //else if (!makingPSF)
          //lix.rwVectorCursor()(i) /= correction(i)*sincConv(i)*sincConv(iy);
          //
          // This without the PS functions
          //
           PBCorrection(i)=FUNC(avgPBVec(i))*sincConv(i)*sincConv(iy);
           if ((abs(PBCorrection(i))) >= pbLimit_p)
                 //lix.rwVectorCursor()(i) /= PBCorrection(i);
            lix.rwVectorCursor()(i) /= sqrt(PBCorrection(i));
           else if (!makingPSF)
                  lix.rwVectorCursor()(i) /= sincConv(i)*sincConv(iy);
        }
```

References

[1] Kemball, A. et al. 2009, "*Calibration and Processing Constraints on Antenna and Feed Designs for the SKA: I*", TDP Calibration and Processing Group Memo 4, http://rai.ncsa.uiuc.edu/CP_Antenna_Feed.pdf.

[2] Smirnov, O.M. 2011, A&A, 527, A107.

[3] Bhatnagar, S. et al. 2008, A&A, 487, 419.

[4] Rau, U. et al. 2009, Proc. IEEE, Vol. 97, No. 8.

[5] Kemball, A. 2008, "*Petascale Computing Challenges for the SKA*", TDP Calibration and Processing Group CPG Memo 1, http://rai.ncsa.uiuc.edu/CPG_1.pdf.

[6] Noordam, J. E. & Smirnov, O.M. 2010, A&A, 524, A61.

[7] Smirnov, O.M. 2011, A&A, 527, A106.

[8] Cortes, G. & Imbriale, W. 2009, "*SKA Offset Optics Design (Progress Report)*", US SKA Technology Development Project Memo 12, http://skatdp.astro.cornell.edu/files/Memo_2009Cortes_OffsetOptics.pdf.

[9] Imbriale, W. et al. 2011, IEEEAC, paper #1597, Version 2, http://ieexplore.ieee.org/stamp/stamp.jsp?arnumber=05747318.

[10] Cornwell, T. et al., 2008, IEEE J. Sel. Topics Signal Process., Vol. 2, No. 5, pp. 647-657.

[11] Chakraborty, N. & Kemball, A. 2009, "*Current state of practice in wide-field, low-frequency, high dynamic range imaging with contemporary radio interferometers*", TDP Calibration and Processing Group CPG Memo 5, http://rai.ncsa.uiuc.edu/CPG_MEMO_5_1.PDF.

[12] Rau, U. 2011, "*Imaging Algorithms in CASA*", http://www.aoc.nrao.edu/~rurvashi/ImagingAlgorithmsInCasa/.

[13] Golap, K. 2010, "*Clean and Imager*", Presentation at CASA Developers Meeting, https://safe.nrao.edu/wiki/pub/Software/CASADevelopersMeeting/imager_dev_2010.pdf.

[14] CASA class hierarchy documentation, http://casa.nrao.edu/active/docs/doxygen/html/hierarchy.html.

[15] Rau, U. 2010, "*Parameterized Deconvolution for Wide-Band Radio Synthesis Imaging*" ,PhD Thesis, 2010, http://www.aoc.nrao.edu/~rurvashi/DataFiles/UrvashiRV_PhdThesis.pdf.

[16] Bhatnagar, S. et al. 2006, "*Correction of errors due to antenna power patterns during imaging*", EVLA Memo 100, http://www.aoc.nrao.edu/evla/geninfo/memoseries/evlamemo100.pdf.

[17] Humphreys, P. & Cornwell, T. 2011, "*Analysis of Convolutional Resampling Algorithm Performance*", SKA Memo 132, http://www.skatelescope.org/uploaded/59116_132_Memo_Humphreys.pdf .

[18] Cornwell, T. J. 2007, "*The Impact of Convolutional Resampling on ASKAP and SKA Phase I Computing Costs*", Technical Report CONRAD-SW_0008, ATNF, KAT-CSIRO, http://www.atnf.csiro.au/projects/mira/documents/CONRAD-SW-0008.pdf.

[19] Smirnov, O.M. 2011, A&A, 527, A108.

[20] Noordam, J. E. 2004, in Ground-based Telescopes, ed. J. M. Oschmann, Jr., SPIE, Conf. Ser., 5489, 817.

[21] Bhatnagar, S. et al. 2004, "Solving for the antenna based pointing errors," EVLA Memo 84, Tech Rep., http://www.aoc.nrao.edu/evla/geninfo/memoseries/evlamemo84.ps.

[22] Cornwell, T. J. & Perley, R. A. 1992, A&A, 261, 353.

[23] Yashar, M. & Kemball, A. 2010, "*Computational Costs of Radio Imaging Algorithms Dealing with the Non-Coplanar Baselines Effect: I*", TDP Calibration and Processing Group Memo 3, http://rai.ncsa.uiuc.edu/SKA/CPG_Memos.html.