

## Executing AntConfigServer

AntConfigServer is a commandline utility that simulates an antenna layout input file according to observation settings and then produces output. The Unix command is of the form:

```
AntConfigServer instructionfilename settingsfilename inputfilename [outputnametag]
```

So a typical call would be:

```
AntConfigServer instruction.txt setting.txt layout.wgs84.txt80x2 output
```

This would read instructions from `instruction.txt` and settings from `setting.txt` and a layout from the file `layout.wgs84.txt80x2` and produce output files prefixed with `output`. If `outputnametag` is not specified it is assumed to be `output`.

A few Matlab scripts are included in the software distribution to show how the information can be extracted and visualized.

## Instruction file

The instruction file is a text file that tells AntConfigServer what to do. An example of an instruction file is given below:

```
uvoptimsigma = 400
nsteps = 10
nlinesteps = 20
lambda = 0
nevaluations = 2
outputflags = 536870911 524288
uvtargetsigmas = 0 400
gridsize = 512
standardpsfpixelpersigma = 10
```

AntConfigServer can perform two basic functions: it can optimize a layout and/or evaluate a layout. The first four parameters in the instruction file relate to optimization settings while the rest of the parameters are associated with performance evaluation. The table below describes the values used:

uvoptimsigma	The baseline sigma value (in meters) of the ideal uv distribution to be optimized towards. Only applicable if nsteps>0.
nsteps	Number of optimization steps to perform. This will result in the antenna positions being modified before the performance of the resulting layout is evaluated.

nlinesteps	Number of linesearch optimization steps used internally (when lambda=0, to find a value for lambda). Can be kept at 20 or reduced slightly for increased speed at a risk of not converging accurately to an optimal value for lambda.
lambda	The fractional magnitude of antenna movement along an optimal gradient direction during optimization step is scaled by lambda. Set lambda=0 to specify that the value for lambda must be determined automatically using a line search. Else set lambda to a fixed value (then nlinesteps is ignored). It is potentially useful to set a fixed small value for lambda when one want to modify a preferred layout only slightly, biasing for better performance for another observation setting.
nevaluations	The number of performance evaluations to be performed for the layout. This MUST correspond to the number of elements in outputflags and uvtargetsigmas.
outputflags	A list of outputflags that are 32 bit integers separated by spaces. These values determine which output files are produced per evaluation (one to one correspondence with uvtargetsigmas). A section below describes the values used for outputflags in detail. Typical values are 536870911 to stipulate that all possible output must be produced or 524288 to write a report only.
uvtargetsigmas	A list of target sigma values for the uv distribution (in meters). Read more about this below. 0 means analyze natural point spread function. >0 means that the uv samples are weighted to achieve this target sigma for uv samples.
gridsize	The number of pixels per dimension used for gridding the uv samples (no unit). This has effect on variables derived from the following outputflags: OUTPUT_GRID_UV, OUTPUT_GRID_RESID, OUTPUT_GRID_UV or OUTPUT_PROFILE_RESID. (See below).
standardpsfpixelpersigma	The PSF in the image domain will be scaled so that one standard deviation of the PSF corresponds to this many pixels (in pixels if no unit).

## Settings file

The settings file is a text file that assigns values to certain parameters used during the simulation. An example of a settings file is given below:

```
settingUID = 0
dishsize = 12 m
declination = -10 deg
reflongitude = 30 deg + 2.5 hrs - 5 min
duration = 4 hrs + 5 min
onsource = 30 min
offsource = 30 min
correlatordump = 5 min
frequency = 1.4 GHz
fracbandwidth = 0.5
nchannels = 1
```

Each of these parameters must be assigned appropriately. The following units can be interpreted (and arbitrarily combined as illustrated above):

"hrs", "hours", "min", "sec", "s", "deg", "degrees", "rad", "radians", "m", "pixels", "pix", "Hz", "MHz", "GHz"

If no unit is given, the default is either radians, pixels, Hz or none depending on the parameter (duration is for example converted to radians from whatever unit you use). An explanation for each parameter follows:

settingUID	Unique unsigned integer identifier for this experiment/observation combination.
dishsize	Dish size (in meters if no unit)
declination	Declination of observation (in radians if no unit)
reflongitude	Reference longitude of observation (in radians if no unit). The timing of the observation is centered around this longitude so that halfway through the observation, the antennas will be pointing to this longitude. If this longitude equals the average longitude of the site, the antennas will be pointing at the zenith (if declination also equals site latitude) halfway through the observation.
duration	The duration of the observation (in radians if no unit).
onsource	The onsource time of the antennas (in radians if no unit). The antennas are observing during onsource time, then is switched off for the duration of the offsource time, and then on again (the cycle repeats), until the duration of the observation has elapsed.
offsource	The offsource time of the antennas (in radians if no

	unit). See onsource.
correlatordump	Every so often the correlator dumps integrated uv samples to be processed (in radians if no unit). Correlator dumping will start (in the observation cycle) when onsource starts irrespective of whether or not it is a multiple of onsource+offsource. So it is as if the correlator is switched off when the antennas are offsource and then restarted with no delay as the antennas become onsource.
frequency	Frequency of the observation (in Hz if no unit). This value has no effect on the results. The results must be scaled by frequency by the user to determine actual values for resolution, for example. Do not call AntConfigServer with different frequency value settings. See below for equations on calculating resolution.
fracbandwidth	Fractional bandwidth for multi-frequency synthesis (no unit). The relative frequency variation over which MFS must be performed. This value has no effect when nchannels=1.
nchannels	The number of channels used for multi-frequency synthesis (no unit). Frequency is varied over fracbandwidth using this many channels.

### Antenna layout input files

The filename is a very important part of the input file specification since it provides header information that is absent in the file. The file contains only data and no header information. There are two filename extensions per filename. The first extension specifies either wgs84 or east north up as the coordinate system for the coordinates in the file. The last filename extension specifies whether the file is binary or text, the number of antennas and the number of columns in the file, which could be 2, 3 or 4.

It is perhaps clearer to go through a few examples:

- **layout.wgs84.txt80x2** : this is a text file with 80 rows and 2 columns containing longitude and latitude coordinates (floating point style) in degrees and spaces between the values. One coordinate per line in the file.
- **loglayout.21E31d1S.txt20x3**: this is a text file with 20 rows and 3 columns containing coordinates east (in meters) north (in meters) and up (in meters) from a reference point specified at 21 degrees East, 31.1 degrees South. Columns are separated by spaces.
- **anotherlayout.10d342W31d14S.txt20x4**: this is a text file with 20 rows and 4 columns containing coordinates east (in meters) north (in meters) up (in meters) and enable (1 for enable, 0 for disable) from a reference point specified at -10.342 degrees East, 31.1 degrees South. Columns are separated by spaces.
- **binarylayout.wgs84.float100x3**: this is a binary file containing a list of

100x3 32 bit single precision floating-point values. Longitude (in radians) Latitude (in radians) Altitude (in meters above wgs84 ellipsoid). The 100 element longitude vector is read first, then the 100 element latitude vector then the 100 element altitude vector.

If there are less than three columns, the altitude is set to 0 meters above the wgs84 ellipsoid. If there are less than four columns then all antennas are enabled

## Output flags

The **outputflags** tell AntConfigServer what to calculate and what output to produce. It may be inefficient to calculate or write out unnecessary information. Outputflags could be any combination of the following values (sum them together, or use the mask **536870911** to calculate everything):

Output flag name	Value	Description
OUTPUT_TIMING	2 <sup>0</sup>	Corresponds to variable <i>timing</i> in Matlab. <i>ntimingsamples</i> output timing data written to [outputnametag]_timing.float[ntimingsamples] in radians.
OUTPUT_PROJ_S	2 <sup>1</sup>	Corresponds to variable <i>proj_s</i> in Matlab. <i>ntimingsamples</i> x <i>nantennas</i> antenna activity state flags per timing written to [outputnametag]_proj_s.uint[ntimingsamples x nantennas]. A variable contains integer flags per timing sample per antenna to indicate if the particular antenna is active or inactive. A value of 0x0001 means that the antenna is disabled. 0x0002 means it is shadowed by earth curvature. 0x0004 means this antenna is shadowed by another antenna. Combinations are also possible, and of course nonzero means inactive.
OUTPUT_PROJ_X	2 <sup>2</sup>	Corresponds to variable <i>proj_x</i> in Matlab. <i>ntimingsamples</i> x <i>nantennas</i> antenna x positions per timing written to [outputnametag]_proj_x.float[ntimingsamples x nantennas] in meters to east (looking from star to earth).
OUTPUT_PROJ_Y	2 <sup>3</sup>	Corresponds to variable <i>proj_y</i> in Matlab. <i>ntimingsamples</i> x <i>nantennas</i> antenna y positions per timing written to [outputnametag]_proj_y.float[ntimingsamples x nantennas] in meters to north (looking from star to earth).
OUTPUT_PROJ_Z	2 <sup>4</sup>	Corresponds to variable <i>proj_z</i> in Matlab. <i>ntimingsamples</i> x <i>nantennas</i> antenna z positions per timing written to [outputnametag]_proj_z.float[ntimingsamples x nantennas] in meters from earth center towards star.
OUTPUT_UV_X	2 <sup>5</sup>	Corresponds to variable <i>uv_x</i> in Matlab. <i>nuvsamples</i> uv sample baseline dx values written to [outputnametag]_uv_x.float[nuvsamples] in meters to east.
OUTPUT_UV_Y	2 <sup>6</sup>	Corresponds to variable <i>uv_y</i> in Matlab. <i>nuvsamples</i> uv sample baseline dy values written to [outputnametag]_uv_y.float[nuvsamples] in meters to north.
OUTPUT_UV_RHO	2 <sup>7</sup>	Corresponds to variable <i>uv_rho</i> in Matlab. <i>nuvsamples</i> uv sample transformed coordinate values written to

		[outputnametag]_uv_rho.float[nuvsamples].
OUTPUT_UV_THETA	2 <sup>8</sup>	Corresponds to variable <i>uv_theta</i> in Matlab. <i>nuvsamples</i> uv sample transformed coordinate values written to [outputnametag]_uv_theta.float[nuvsamples].
OUTPUT_UV_W	2 <sup>9</sup>	Corresponds to variable <i>uv_w</i> in Matlab. <i>nuvsamples</i> uv sample weights written to [outputnametag]_uv_w.float[nuvsamples].
OUTPUT_UV_CX	2 <sup>10</sup>	Corresponds to variable <i>uv_cx</i> in Matlab. <i>nuvsamples</i> uv sample baseline values written to [outputnametag]_uv_cx.float[nuvsamples] in meters to east. These are the uv coordinates corresponding to the centroid of each uv area.
OUTPUT_UV_CY	2 <sup>11</sup>	Corresponds to variable <i>uv_cy</i> in Matlab. <i>nuvsamples</i> uv sample baseline values written to [outputnametag]_uv_cy.float[nuvsamples] in meters to north. These are the uv coordinates corresponding to the centroid of each uv area.
OUTPUT_GRID_PSF	2 <sup>12</sup>	Corresponds to variable <i>grid_psf</i> in Matlab. <i>gridsize x gridsize</i> psf pixel values written to [outputnametag]_grid_psf.float[gridsize]x[gridsize].
OUTPUT_GRID_RESID	2 <sup>13</sup>	Corresponds to variable <i>grid_resid</i> in Matlab. <i>gridsize x gridsize</i> psf residual pixel values written to [outputnametag]_grid_resid.float[gridsize]x[gridsize].
OUTPUT_GRID_UV	2 <sup>14</sup>	Corresponds to variable <i>grid_uv</i> in Matlab. <i>gridsize x gridsize</i> gridded uv weighted (if not natural weighting) count values written to [outputnametag]_grid_psf.float[gridsize]x[gridsize].
OUTPUT_SHADOW	2 <sup>15</sup>	Corresponds to variable <i>shadow</i> in Matlab. <i>nantennas</i> lost uv sample count per antenna written to [outputnametag]_shadow.uint[nantennas]. Record is kept of the number of <i>lost uv samples</i> per antenna. This value per antenna is stored in the variable <i>shadow</i> . For a particular layout, for a particular experiment, this shows which antenna is least useful. These values could be accumulated over many experiments to compare which layout causes the least shadowing for the range of experiments.
OUTPUT_UVDIST	2 <sup>16</sup>	Corresponds to variables <i>uvmaxdists</i> and <i>uvrmsdists</i> in Matlab. <i>nantennas</i> maximum and rms differences between uv samples and their centroid positions per antenna are written to [outputnametag]_uvmaxdist.float[nantennas] and to [outputnametag]_uvrmsdist.float[nantennas].
OUTPUT_COORDINATES	2 <sup>17</sup>	Corresponds to variables <i>longitude</i> and <i>latitude</i> in Matlab. <i>nantennas</i> longitude and latitude coordinates are written to [outputnametag]_longitude.float[nantennas] and to [outputnametag]_latitude.float[nantennas].
OUTPUT_PROFILE_RESID	2 <sup>18</sup>	Corresponds to variables <i>residualmins</i> , <i>residualmaxs</i> , <i>residualsse</i> and <i>residualssen</i> in Matlab. <i>gridsize</i> psf error statistics values (min error, max error, sum square error and pixel counts) are written to [outputnametag]_residualmin.float[gridsize], [outputnametag]_residualmax.float[gridsize],

		[outputnametag]_residualsse.float[gridsize] and [outputnametag]_residualssen.float[gridsize]. These gridded dimensions correspond to that of grid_psf. As the radius is increased around the origin of the PSF more pixels are added into the calculation up to the maximum radius of gridsize/2. Note that the peak sse value is seldom reached when the entire field of view is included. These output vectors allows one to investigate if the sidelobe errors are big closer or further away from the origin.
OUTPUT_WRITE_REPORT	2^19	Writes report to [outputnametag]_report.txt which is a summary of the important results of the simulation.
OUTPUT_WRITE_SETTINGS	2^20	Writes the interpretation of the input settings file to [outputnametag]_settings.txt. This is useful because the original settings file can contain complicated additions using different units which Matlab doesn't understand.
OUTPUT_WRITE_INPUT	2^21	Write the input to [outputnametag]_wgs84.txt[nantennas]x3
OUTPUT_WRITE_INSTRUCTION	2^22	Write the instruction to [outputnametag]_instruction.txt
OUTPUT_ALL	2^29 - 1	Ensures that all possible output is produced.

## Report file

The report file (which is produced when OUTPUT\_WRITE\_REPORT is enabled in outputflags) is the most important summary of results of the simulation. A typical report file could be as follows:

```

settingUID = 0
uvtargetsigma = 0
uvmajor = 393.757
uvminor = 396.559
uvangle = -31.8511 deg
uvminarea = 0.000435789
uvmaxarea = 0.0426426
uvrmsarea = 0.00462203
uvsensitivity = 0.0384692
uvmaxdist = 78.9804
uvrmsdist = 14.3804
residualmin = -0.0449086
residualmax = 0.0407482
residualrms = 0.0146101
ntimingsamples = 4
nuvsamples = 1520
nuvsamplescoincident = 0
nuvlocationcoincident = 0
nuvsamplesexceedbounds = 0
nuvsamplesdisabled = 0
nuvsamplesshadowedearth = 0

```

```

nuvsamplesshadowedantenna = 0
minbaseline = 65.1351
maxbaseline = 1175.08
nantennas = 20
gridsize = 512
standardpsfpixelpersigma = 10
outputflags = 536870911

```

A description of the fields follow:

settingUID	The unique (this is not required to be unique) identifier specified by the user in the settings file for the particular observation settings used to produce this report.
uvmajor, uvminor, uvangle	<p>The major, minor sigma values (in meters) and rotational angle of the UV distribution. This is the Gaussian that has the same curvature at the origin as UV distribution. This is calculated using the singular value decomposition of the covariance matrix of the (weighted) uv samples. The PSF full width half maximum resolution can be calculated using these results as follows:</p> <pre> sigma2fwhm=2*sqrt(2*log(2));%2.35482004503095 rad2arcsec=180*60*60/pi; lambda=299792458.0/frequency; uv2fwhm=sigma2fwhm*rad2arcsec*lambda*2/pi; psfmajor=uv2fwhm/uvminor; psfminor=uv2fwhm/uvmajor; psfangle=(uvangle+pi/2)*180/pi; </pre> <p>Note that this resolution corresponds to the best curvature fit around the origin only and not the best fit to the PSF including image domain sidelobes. When <code>uvtargetsigma&gt;0</code> then weighting of uv samples takes place to try achieve the best fit target resolution (including sidelobe effects). In this case the output best fit PSF FWHM resolution becomes:</p> <pre> psftargetfwhm=uv2fwhm/uvtargetsigma; </pre> <p>which may differ slightly from the origin curvature solution <code>psfmajor</code>, <code>psfminor</code>, <code>psfangle</code> given above.</p>
uvminarea, uvmaxarea, uvrmsarea	These output variables provide statistics of the Voronoi areas calculated for the uv samples given the transformation specified using <code>uvtargetsigma</code> .
uvsensitivity	<p>Sensitivity is the fraction of the noise received by the antenna that comes through to the image.</p> <pre> uvsensitivity=sqrt(sum(uv_w.^2))/sum(uv_w); </pre> <p>The natural sensitivity corresponds to natural weighting (when all measured UV samples are weighted equally):</p> <pre> Naturalsensitivity=1/sqrt(nuvsamples) </pre>

uvmaxdist, uvrmsdist	The maximum distance (or the rms distance for all) in the uv domain (meters) a uv sample need to move to reach its area centroid position.
residualmin, residualmax	The overall maximum or minimum residual value.
residualrms	The maximum residual rms value by varying the number of pixels included in the calculation (by increasing the diameter about the origin).
ntimingsamples	The number of correlator dumps that occurred during the observation.
nuvsamples	The number of uv samples that was acquired during the observation. Some uv samples may have been discarded or lost because of shadowing due to other antennas, the earth horizon or because some antennas were disabled by the user (eg for robustness testing).
nuvsamplescoincident	The number of uv samples that coincided with other uv samples exactly at the same points in the uv domain with floating point precision. This is different from gridding points to the same cell in the gridded grid_uv variable.
nuvlocationcoincident	The number of locations in uv space where uv samples exactly coincides. Such points may be useful for auto-calibration purposes.
nuvsamplesexcludedbounds	This number of uv points was excluded from grid_uv during the gridding process because gridsize is too small.
nuvsamplesdisabled	The number of uv points that is lost because some antennas were disabled by the user.
nuvsamplesshadowedearth	The number of lost uv points due to shadowing of the earth's horizon. For this calculation, simply the sign of proj_z is checked. This is correct if the earth was a sphere, but since it is an ellipsoid this is only an approximation.
nuvsamplesshadowedantenna	The number of lost uv points due to shadowing by other antennas.
minbaseline	The minimum projected baseline. Note that this value may be smaller than the minimum dish size when more than one frequency channel is used.
maxbaseline	The maximum projected baseline.
nantennas	The number of antennas in the layout.
gridsize	Value of gridsize recorded in report.
standardpsfpixelpersigma	Value of standardpsfpixelpersigma recorded in report. See instruction file above.
outputflags	The outputflags that specify which output structures were written. See instruction file above.

## Discussion on calculation of UV areas

A region around each uv sample is defined and the weighting factor assigned to the uv sample equals the inverse of this area.

In order to determine the regions, the uv points are first nonlinearly transformed into a new plane (rho theta plane). In the new plane, the uv samples are supposed to have equal area if the distribution of uv samples were perfectly what we wanted.

So, if our ideal distribution is a Gaussian of standard deviation sigma, we transform the uv points through the Cumulative Distribution Function of that ideal Gaussian. If our distribution of uv points matched the ideal Gaussian, then after mapping the points through the CDF we should have a perfectly uniform distributed points where each point is equally spaced from its neighbours so that the areas attributed to each point are equal. If our distribution of points did not match the ideal Gaussian then some points may be closer together and we will not have a uniform distribution, and the areas attributed to each point will not be equal.

We use Voronoi tessellation to determine the boundaries between distinct uv points, defining areas attributed to each distinct uv sample. Those uv samples that are exactly coincident in the uv plane divide the area of such a region equally amongst each other.

It is important to consider boundary conditions so that we calculate areas of the finite space of a uniform distribution. Classic Voronoi tessellation considers an infinite space resulting in infinite areas for boundary points.

In practice we use a polar coordinate system rather than Cartesian, for three reasons: 1) the polar CDF of a Gaussian is numerically quicker to compute than its Cartesian equivalent, Erf(). 2) for logarithmic distributions we must use polar coordinates because negative numbers are not allowed. 3) only one dimensional boundary conditions need to be evaluated (for rho) because theta repeats and uv samples are duplicated 180 degrees apart (then we can overwrite values close to the branch cut with central values).

The polar CDF of a Gaussian is:

$$\text{GaussianCDF}(r) = \text{Integral}(0, r, t \cdot \exp(-0.5 \cdot t^2 / \sigma^2))$$

$$\text{so normalizedGaussianCDF}(r) = (1 - \exp(-0.5 \cdot r^2 / \sigma^2))$$

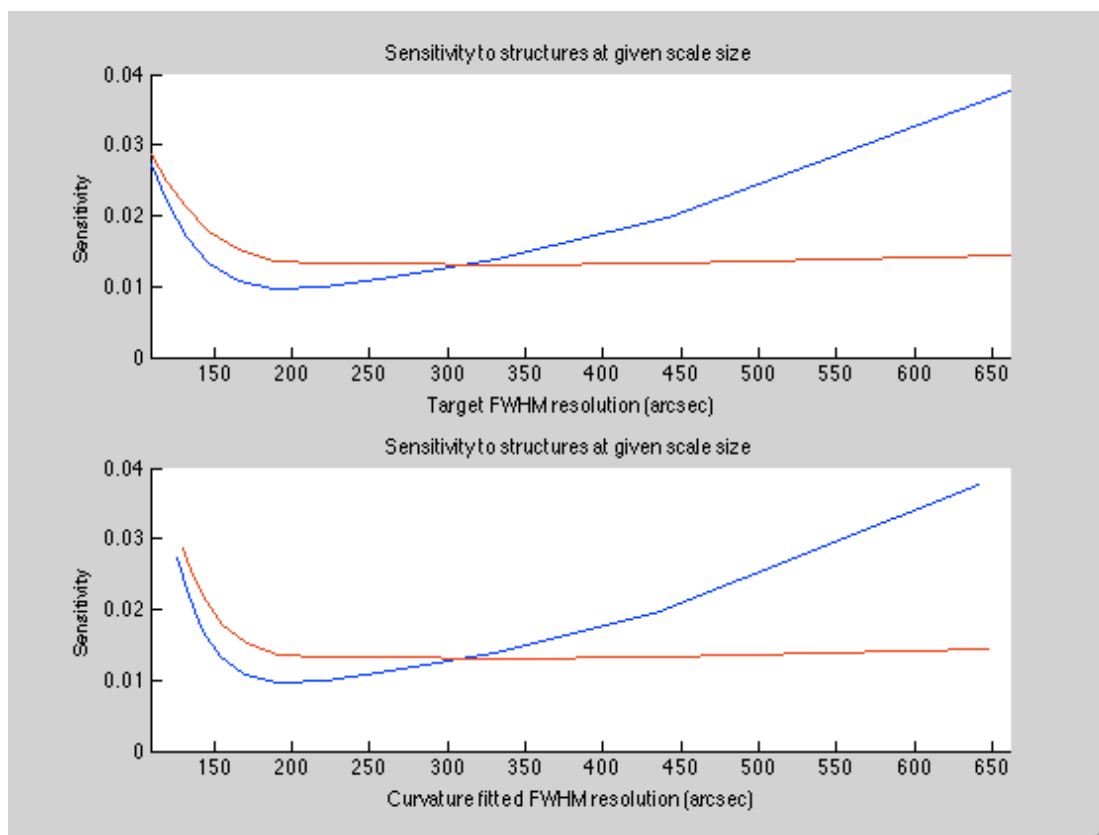
$$\text{with GaussianCDF}(\theta) = \theta$$

I intend to write a paper on this, so a further discussion will follow.

## Discussion on important evaluation parameters

### Uvsensitivity

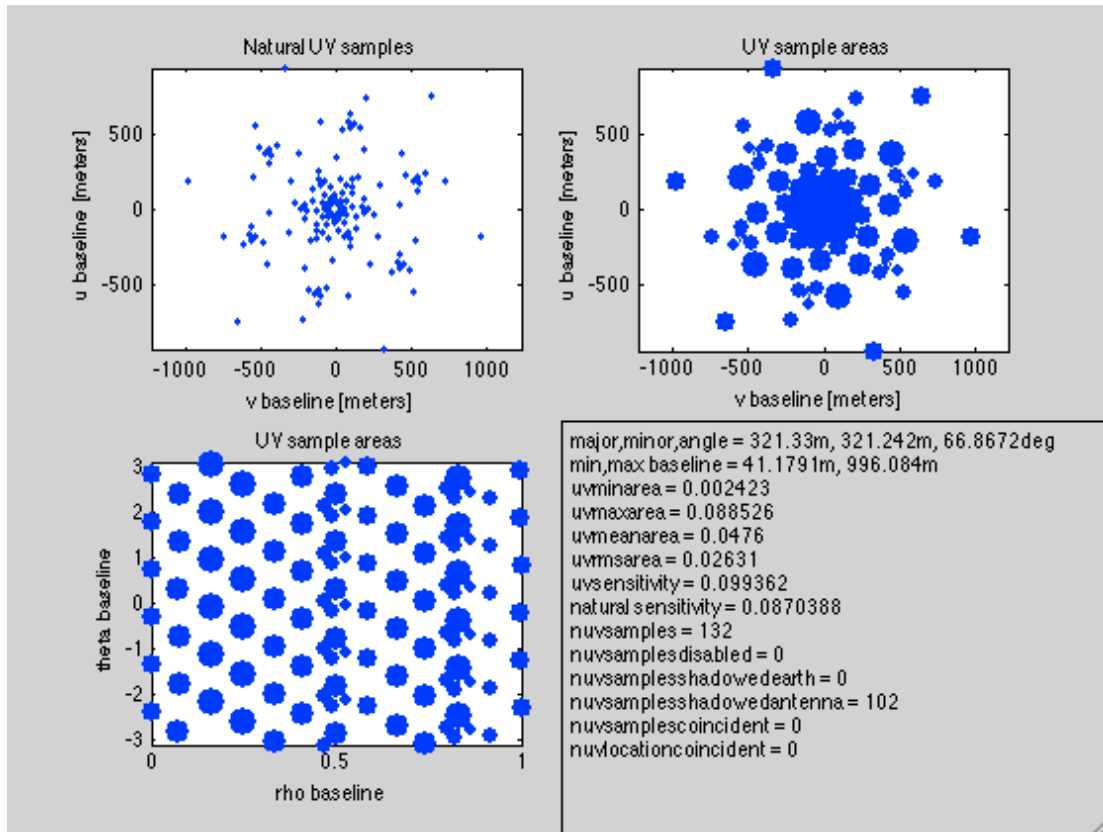
Uvsensitivity is the fraction of noise collected at the receiver that propagates through to the image. For comparative evaluation of different layouts one can plot the uvsensitivities as a function of resolution to see how much better one layout performs than another at different resolutions. Figure 1 shows results comparing a log-spiral to a Gaussian layout with the same number of antennas. The Gaussian layout has better sensitivity to objects of the same scale as its natural resolution (around 180 arcsec, here), than the log-spiral layout. However the log-spiral layout has a nearly constant sensitivity to a much larger range of resolutions before its sensitivity deteriorates.



**Figure 1:** Sensitivity versus resolution results for log-spiral (red) compared to Gaussian (blue) layouts.

In case one wish to summarize the sensitivity performance (which is really a function of resolution) as a single value, then one can use the uvsensitivity that is calculated when  $uv_{\text{targetsigma}}=0$ . In this case the uv weights are calculated to achieve a logarithmic distribution instead of a Gaussian of specified  $uv_{\text{sigma}}$ .

The subplot in the bottom left corner of figure 2 shows how the uv points of a log spiral is transformed into the rho x theta space (using  $uvtargetsigma=0$ ) such that they are nearly equally separated.

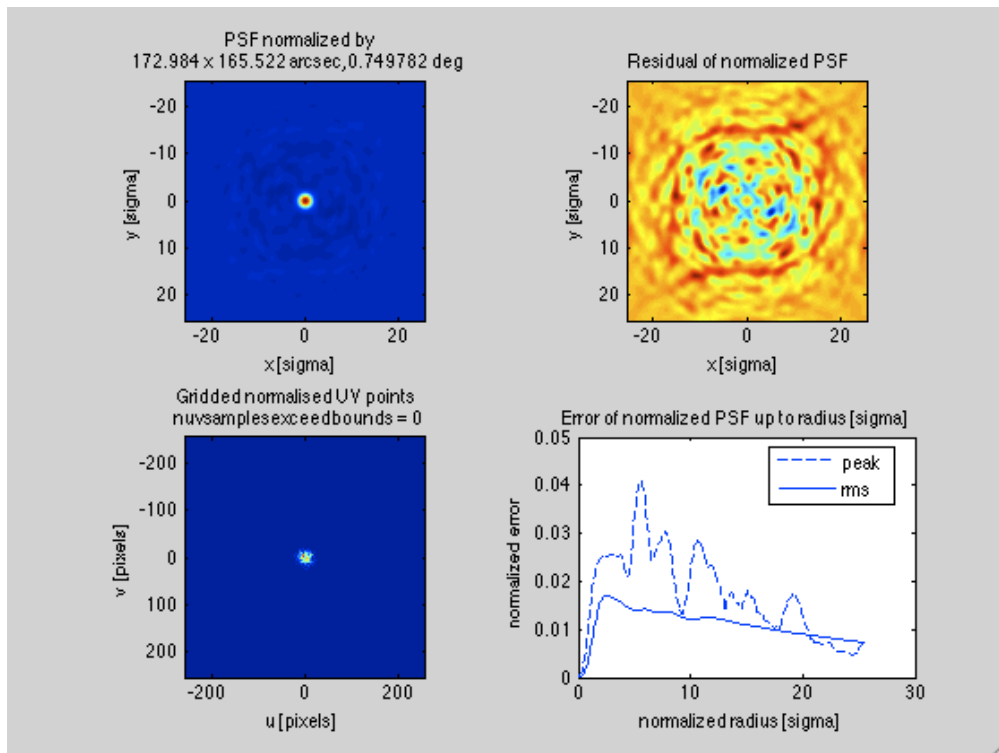


**Figure 2:** UV snapshot results for a log-spiral

## Peak rms residual error

The residual error is of particular interest because the magnitude of the errors and the sidelobes particularly can be intuitively interpreted to gauge how good a PSF is. Generally it is difficult to get a robust measure for calculating the PSF residual because it is dependent on many gridding options. The approach followed here chooses gridding options so that comparisons are as independent from the gridding options as possible. This is done by normalizing the PSF prior to the residual calculations. The PSF is rotated and scaled by the inverse of its major and minor axes so that it is gridded without units in terms of pixels per sigma (standard deviation). A circularly symmetric PSF is therefore achieved (at least in terms of the curvature of the PSF near the origin). Before the primary beam power pattern is multiplied to the PSF, it too is appropriately rotated and scaled (a Bessel function related to the dish size is used to approximate the voltage pattern).

The output profiles `residualpeak` (from `residualmins`, `residualmaxs`), `residualsse` and `residualssen` are produced. As the radius around the origin increases, more pixels are included in the calculation of the peak residual error, or the rms residual error (from `residualsse` and `residualssen`). At some point, around 2.4 sigma in the case of figure 3, the largest accumulated residual error is achieved. This means that when we accumulate the squared residual errors from radius zero up to 2.4 sigma, in this case, (and divide by the number of errors summed, and take the square root) the biggest rms error is achieved. This value seems to be of most robust significance when comparing the performance of layouts in terms of the image domain.



**Figure 3:** Results showing residual error profiles accumulated up to a certain radius in the bottom right corner.