National Radio Astronomy Observatory

Socorro, N.M.

VLA Scientific Memorandum No. 154

THE WEIGHTING OF VISIBILITY DATA

R.J. Sault
August, 1984

1. INTRODUCTION

In making maps, a weight is usually applied to each gridded visibility cell to compensate for the local density of ungridded visibilities. Optimum signal to noise ratio dictates constant weighting (natural weighting). In cases where the noise is less significant than sidelobe levels, minimizing the rms sidelobe within the dirty map or beam leads to weights which are inversely proportional to the local density function (uniform weighting).

Though the deconvolution step should, in principle, remove sidelobes regardless of the weighting scheme used, the practicalities are quite different. The CLEAN algorithm tends to be faster and more reliable when the sidelobe levels in the beam are low (this helps avoid the corrugation effect). The maximum entropy method poorly deconvolves sidelobes imbedded in extended structure, and the Cornwell and Evans algorithm (VM) converges poorly when the dirty beam is broad (In determining the inverse Hessian, VM approximates the dirty beam by a delta function. Though an iterative improvement technique has been proposed, by Cornwell, to improve the estimate of the inverse Hessian, this costs extra FFT's). Heuristically most deconvolution algorithms should work better, when the problem they are given is simpler.

We must not loose sight, however, of what weighting cannot do. Weighting is essentially a simple linear deconvolution technique. It cannot interpolate or extrapolate unsampled visibilities, and consequently some sidelobes are fundamentally unremoveable by weighting. For a well filled in u-v coverage, little improvement over uniform weighting can be gained. Other weighting schemes are useful only when the u-v coverage contains significant holes, as in a VLA snapshot.

In summary, by appropriate weighting the quality of both the dirty and deconvolved map can be improved. This memo considers weighting schemes in general and algorithms to calculate optimum weights. Examples and results are also given. An appendix describes an AIPS task used to implement two weighting schemes.

2. WEIGHTING IN GENERAL

Ignoring a scaling constant used to make the beam peak unity, then natural and uniform weights are defined as:

$$W(\ell) = M(\ell)$$

and

$$W(\ell) = M(\ell)/S(\ell)$$

respectively. One dimensional notation will be used for simplicity, as extension to higher dimensions is straight forward. Here $W(\ell)$ is the weighting function and $S(\ell)$ is the sampling density function (measured in visibilities per grid cell). $M(\ell)$ is the transform of the ideal beam. Though the ideal beam is usually a delta function (so $M(\ell)$ will be constant), it can be made a gaussian to both suppress sidelobes (at the expense of resolution) and improve brightness sensitivity to extended objects.

Defining the grid function, $B(\ell)=W(\ell)S(\ell)$, as the transform of the dirty beam, then for natural and uniform weighting, the grid function at sampled cells is:

$$B(\ell) = S(\ell)M(\ell)$$

and

$$B(\ell) = M(\ell)$$

respectively, and zero at unsampled cells. A general description of weighting is given by Sramek (1982).

It is easily shown that uniform weighting minimizes the rms sidelobe in both the map and beam. However the actual sidelobe level will be strongly dependent on the size of the field of view (or alternately the coarseness of the gridding in the u-v plane). As the field increases in size, the u-v plane grid becomes finer, until there is only one visibility per u-v cell, and the sampling density function reduces to zeros or ones. Thus uniform weighting has degenerated into natural weighting, with the accompanying high sidelobes. As Clark notes (1979) this is in a sense fundamental, yet in another sense only apparent, if the sky is assumed mostly blank.

To make a map of a large field but suppress sidelobes over a smaller field, the uniform weights of the small field can be used on the large field map. This is essentially what super-uniform weighting does. More specifically, the super-uniform weight is given by:

$$W(\ell) = M(\ell)/(S(\ell)*\Pi(\ell))$$

Here $\Pi(\ell)$ is a box function (typically 3-9 cells wide), and "*" represents convolution.

Generally we want to suppress sidelobes in the beam over a region where the source's autocorrelation is non-zero. This prevents sidelobes from one region of emission falling onto other regions of emission. An optimal approach would be to find weights which minimizes sidelobes in a given window. Additionally we could give different importance to sidelobes in various regions. This leads to minimizing a weighted-squares measure of sidelobe level, viz:

$$\varepsilon^2 = \Sigma_k q(k)(b(k)-m(k))^2$$

Here q(k) is the window function, choosen to be large where sidelobe suppression is important, and small or zero elsewhere. Also b(k) and m(k) are the transforms of $B(\ell)$ and $M(\ell)$ respectively. By differentiating and equating to zero, the optimum grid function is found to satisfy:

$$Q(\ell)*B(\ell) = Q(\ell)*M(\ell) \qquad \ell \in \{ \text{ sampled cell } \}$$

Here $Q(\ell)$ is the transform of q(k). If q(k) is constant, then $Q(\ell)$ is a delta function, and the solution is uniform weighting, as expected.

3. ALGORITHMS

Though the above equation resembles a deconvolution problem, it is different in that the equality between sides only holds at sampled cells (not the entire plane!). Inverse or Weiner filter approaches are neither of theoretical or practical use in its solution.

Two algorithms for solving this have been investigated. Firstly the classical van Cittert algorithm (Schafer et al, 1981) can be used. An iteration of this algorithm is defined as:

$$B_{n+1}(\ell) = B_n(\ell) + \lambda(Q(\ell)*M(\ell) - Q(\ell)*B_n(\ell)) \qquad \ell \in \{ \text{ sampled cells } \}$$

Here λ is a constant in the range

$$0 < \lambda < 2/q_{max}$$

(q_{max} being the maximum of the window function). The algorithm is guaranteed to converge to a unique solution if the window function is everywhere positive and non-zero. If the window function is zero in some regions, then the iteration will converge to a minimum norm solution for

$B(\ell)$ (provided B_Q is zero), but the solution need not be unique. The attraction of the algorithm is that it is reliable and well understood. It is, however, slow to converge (typically taking 20-30 iterations), and quite expensive computationally (2 FFT's per iteration). We will call the weights produced by this algorithm "optimal weights". The algorithm is too expensive to be of practical use, but has been used here to enable comparisons with approximate solutions.

An alternate algorithm is derived as follows: For any B':

$$Q(\ell)*(B'(\ell)B(\ell)/B'(\ell)) = Q(\ell)*M(\ell)$$

If $B'(\ell)$ is a good estimate of the grid function, then $B(\ell)/B'(\ell)$ will be roughly unity. Then

$$Q(\ell)*(B'(\ell)B(\ell)/B'(\ell)) \sim (Q(\ell)*B'(\ell))B(\ell)/B'(\ell)$$

Therefore

$$B(\ell) \sim B'(\ell)(Q(\ell)*M(\ell))/(Q(\ell)*B'(\ell))$$

It has been found in practice that the approximation is quite good when:

1) Q is reasonably narrow (q reasonably broad), and/or
2) Q is positive, which will occur if q is an autocorrelation function. Additionally if q is positive (which a window function should be), the approximation appears to be very good, regardless of the broadness of Q.

This suggests an iterative scheme, where B' is the grid estimate from the previous iteration. However it seems that 1 iteration is usually enough in practice. Possible estimates for $B'(\ell)$ are the natural, uniform or super-uniform grid functions. Uniform and super-uniform grid functions should provide better results than the natural grid function. We will call weights produced by this equation "approximately optimal weights".

Note that making $B'(\ell)$ the natural grid function, making $Q(\ell)$ a box function and assuming that the ideal beam is a delta function, then approximately optimal weighting reduces to super-uniform weighting (if the ideal beam was not a delta function, then approximately optimal and super-uniform weighting will be slightly different).

The approximations made suggest that a superior super-uniform-like weighting scheme should be achieved by making B' the uniform grid function. This produces weights proportional to the uniform weight, divided by the number of sampled grid cells in some neighbourhood. Note that when the u-v plane is completely sampled, then this scheme will always lead to uniform weighting (which is always optimal in this case) whereas super-uniform will not. The scheme would be very slightly simpler and cheaper computationally than super-uniform weighting. We shall call this "averaged-uniform weighting".

Finally note that as Q in super-uniform weighting is a box function, q (a sinc) will be a somewhat queer window function, since it will have negative parts.

4. RESULTS

This section describes a myriad of beam patterns produced by different weighting schemes (natural, uniform, super-uniform, optimal and approximately optimal). Measures of sidelobe levels are also given.

All beam patterns come from a 4 minute snap-shot of the VLA. The u-v data base contained 4465 visibility records, the maximum baseline was 160 kilolambda, and the u-v cell size was 1.6 kilolambda. No tapering was used (i.e. the ideal beam was a delta function). All contour maps show contours at the -10,-5,5,10,25,50,75 and 100 percent level. The measure of sidelobe level used will be the weighted-squared sidelobe level, ε^2, normalized by the weighted-squared sidelobe level of the optimum solution.

Figures 1 and 2 show the natural and uniform beams. The sidelobe power (not rms) is 34.3% worse in the natural beam than in the uniform beam.

4.1 Super-Uniform and Sinc-Lobe Window Functions

A comparison between super-uniform and other weighting strategies is clearly called for. However a direct comparison (at least with optimal weighting) is not possible as the super-uniform window function (a sinc) has negative regions. A reasonable comparison will be gained by using only the main (positive) lobe of the sinc. Thus beams were produced for:-

a) Optimal weighting, using the main sinc lobe as window function.

b) Approximately optimal weighting, using either the full sinc, or just its main lobe. The initial estimate of the beam was either natural (which would produce super-uniform beams, when the full sinc is used), uniform, or a previously generated approximately optimal beam (i.e. use iteration).

Figure 3 shows the super-uniform beams corresponding to averaging neighbourhoods of 3x3, 7x7, 11x11 and 15x15 u-v cells. The relation between the region of sidelobe suppression and the size of the sinc function (main lobe size is given on the side of contour diagrams) is clear for the first three cases. However the fourth example shows a worsening in near in sidelobes, indicating that the approximation used in deriving super-uniform weighting is bad for this case.

For the 3x3 case, super-uniform weighting has an ε^2 only 5.8% worse than the optimum (A uniform beam had an ε^2 20.1% worse). However as the averaging neighbourhood increased, super-uniform weighting became further from optimal (Optimal and super-uniform were still reasonably close for the 11x11 case). Super-uniform beams usually had more sidelobes outside the suppression region than did the optimal beams (compare Figures 3c and 4 for the 11x11 case).

Averaged-uniform weighting was superior to super-uniform weighting in all cases tested. Again, however, the superiority was not large, but increased as the neighbourhood size increased (For the 3x3 case, averaged-uniform had an ε^2 1.9% better, whereas for the 11x11 case, it was 8.0% better).

Iterating using the approximately optimal approach did show a steady decrease in ε^2. The results of the first iteration were, however, so close to optimal, that the improvement was not significant.

4.2 Large Disc

Figure 5 shows a window function (roughly a large disc) and its resulting optimally weighted beam. The approximately optimal beams, using the natural and uniform beams as the initial estimates, were both quite similar to the optimal, having ε^2 8.5% and 5.7% worse. The plain uniform beam's ε^2 was 18.7% worse.

4.3 Medium Disc and Two Small Disc

Figures 6 and 7 show two different window functions and their corresponding weighted beams. The approximately optimal algorithm produced essentially rubbish (as Q has negative regions, the algorithm leads to a near divide by zero, resulting in the beam being dominated by a few visibilities). Both these cases the window functions are becoming more specialized, and the resulting sidelobe suppression more impressive. Figure 7 is particularly impressive, with large sidelobes being suppressed, at the expense of new large sidelobes in the map center. For this case ε^2 has been decreased by over seven-fold relative to the uniform beam.

4.4 Autocorrelations as Window Functions

As mentioned, beam sidelobes only need to be suppressed in the non-zero region of the sources autocorrelation. Two tests were made with a field which contained two point sources, positioned to be in the worst sidelobes of each other. The window functions used were:

1) the autocorrelation of the CLEAN image, and

2) the autocorrelation of two boxes which contained the point sources.

Figure 8 shows the results for case 2.

In both cases it was found that the approximately optimal algorithm worked well. Indeed for case 1 the optimal algorithm never converged satisfactorily (too slow) and the approximately optimal solution was the best result. Additionally using the uniform, rather than the natural, beam as the initial estimate produced much better results. For example, for case 2 using a uniform beam as the initial estimate gave ε^2 only 0.2% worse than optimal, whereas for the natural beam, ε^2 was 24.8% worse.

Some general comments should be made. Firstly (as has been previously mentioned) the approximately optimal algorithm works well with autocorrelations as window functions (especially when the initial beam is uniformly weighted). Secondly autocorrelations will always have their maxima at the center of the field. As the beam sidelobes are always high near the center, the resulting beams tend to give narrow central spikes but not-as-good suppression elsewhere. A comparison of Figures 7 and 8 illustrates this. To get good localized sidelobe suppression (as in Figure 7), neither an autocorrelation as window function, nor the approximately optimal algorithm, can be used.

5. WEIGHTING TO IMPROVE THE DIRTY MAP

This memo concentrates on minimizing sidelobes in the dirty beam. This is partially for simplicity, and partially because it is the sidelobes of the dirty beam which create problems for the deconvolution algorithms. An alternate approach would be to minimize the sidelobes in the dirty map. A derivation indicates that the weights satisfy:

$$Q(\ell)*(W(\ell)I(\ell)) = Q(\ell)*I(\ell) \qquad \ell \in \{\text{ sampled cells }\}$$

(Here $I(\ell)$ is the transform of the source, $i(k)$). To solve for the weights we would need an initial model of the source, as with self-cal (An accurate model would be far less critical, and easier to determine, than it is with self-cal). In addition if $q(k)$ or $i(k)$ were not even, then the resulting weights would generally not be real, and the beam would not be even (Note, however, if Q where a delta function, i.e. uniform window function, the the $I(\ell)$'s cancel, and we return to uniform weighting). Constraining the weights to be real complicates the above equation to:

$$\mathrm{Re}(I(\ell)(Q(\ell)*(W(\ell)I(\ell)-I(\ell)))) = 0 \qquad \ell \in \{\text{ sampled cells }\}$$

(Here Re(x) is the real part function). This appears a somewhat difficult equation to solve (even approximately). The additional complication, and the probable little extra gain in sidelobe suppression make this approach unattractive.

6. CONCLUSIONS

Roughly speaking, we can weight to suppress sidelobes
1) in the dirty beam, in a region near the beam center, or
2) in the dirty beam, where the autocorrelation of the source is non-zero, or
3) in the dirty map.

The first approach tends to be of general application, whereas the others are more data dependent and specialized. The first approach produces pretty good results, whereas the gains achieved by the second approach do not justify the added complexity (For example compare Figures 3a and 8b near the beam center). Extrapolating, the same can be said for the third case. There may, of course, be specific examples where the latter two approaches may be useful.

Super-uniform weighting was found to be, in a sense, nearly optimal at suppressing sidelobes in a region near the beam center, provided the averaging neighbourhood was less than about 9x9 cells. Averaged uniform weighting (a simple modification of super-uniform) works slightly better for small averaging neighbourhoods, and can work substantially better for large averaging neighbourhoods. The latter weighting scheme should be made an option in the map making programs.

One question not so far considered is: "Are there better Q functions for general suppression than the box function of super-uniform and averaged uniform weighting?" For practical implementation Q must have finite support, and q must be centralized. These are the same requirements as for the gridding convolution function. However the requirement that q is centralized is less important than in the gridding case. Additionally to make the approximately optimal algorithm work well, Q and q should both be (essentially) non-negative. One desirable function would seem to be a gaussian truncated in the u-v plane.

7. REFERENCES

Clark B.G., (1979), Digital Processing Methods for Aperture Synthesis Observations, in Image Formation from Coherence Functions in Astronomy, ed C. van Schooneveld.

Schafer R.W., Mersereau R.M., Richards M.A.,(1981), "Constrained Iterative Restoration Algorithms", Proc IEEE, v69,n4,p432-450.

Sramek R.A., (1982), Map Plane - u,v Plane Relationships, in Synthesis Mapping, ed A.R. Thompson, L.R. D'Addario.

8. APPENDIX - AIPS TASK UVWT

UVWT determines the weights for visibility data to minimize a weighted error measure. This is a highly experimental. There are two possible algorithms, the 'optimal' and 'adaptive' (approximately optimal) algorithms.

a) The optimal algorithm uses a van Cittert iteration, and takes about 60 FFT's (30 iterations). It is of little practical use, other than for comparison in tests.

b) The adaptive (approximately optimal) algorithm uses a technique similar to super-uniform weighting. This takes 3 FFT to perform.

The algorithm to use is given by OPCODE, values being 'OPTI' and 'ADAP'.

Two input files are required by the task, the beam file and the window file. The beam file can be a file which has been formed with any sort of weighting. It should not, however, be grid corrected. For the optimal algorithm, the type of weighting of the input beam does not effect the output in any respect. However for the adaptive algorithm, the input beam should be as close to the desired beam as possible (i.e. use uniform or super-uniform weighting, or possible a beam obtained from UVWT. If the output beam is to be tapered, so should the input).

The window file indicates the penalty for sidelobes, in a weighted-least-squares sense. Generally it will be large in the central regions, and small or zero at the edges. It must be even and centered on (N/2,N/2+1) (like the beam).

The output file will be a "weights file" to apply to the map and beam. They are applied by convolving. DO NOT USE CONVL to do this (CONVL gets it wrong), but use the tried and true approach of FFT(weight file), FFT(map), COMB(multiply real part), COMB(multiply imaginary part), FFT. Also the grid correction will have to be done manually (good luck!).

APARM gives some extra parameters. Because of rounding noise, UVWT has to use a threshold below which it realizes a cell has not been sampled. This is set as APARM(1) by abs(min value). As the minimum value will be negative and due to rounding noise, the threshold estimate is quite good. APARM(1) is, by default, 75, and should not need to be changed. A warning message is given if UVWT thinks the threshold is bad.

The taper to apply to the data is given by APARM(2), in arcsec. For the optimal algorithm, it does not matter if the input beam is tapered. But generally for the adaptive algorithm, the input beam should be as good an approximation of the desired beam as possible.

APARM(3) and APARM(4) are used, for the optimal algorithm, to determine when to stop iterating. APARM(3) is the maximum iteration count. If the relative rms change drops below APARM(4), then the iteration also stops. Good values are 30 iterations and relative rms change of 0.001.

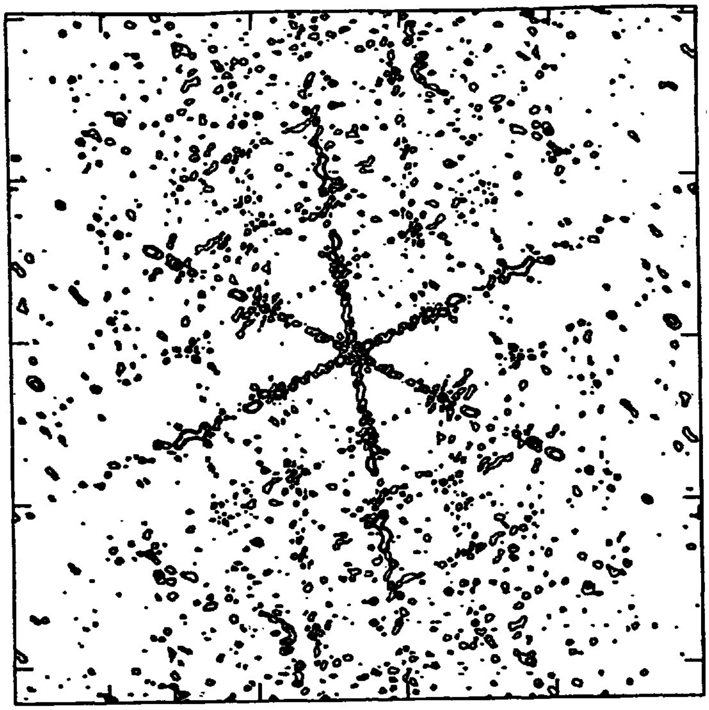

Figure 1
Uniform Beam

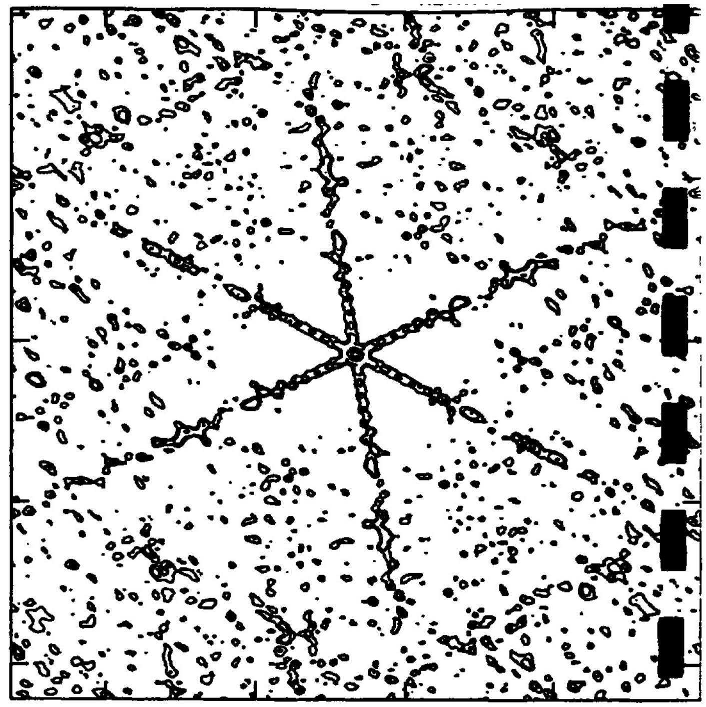

Figure 2
Natural Beam

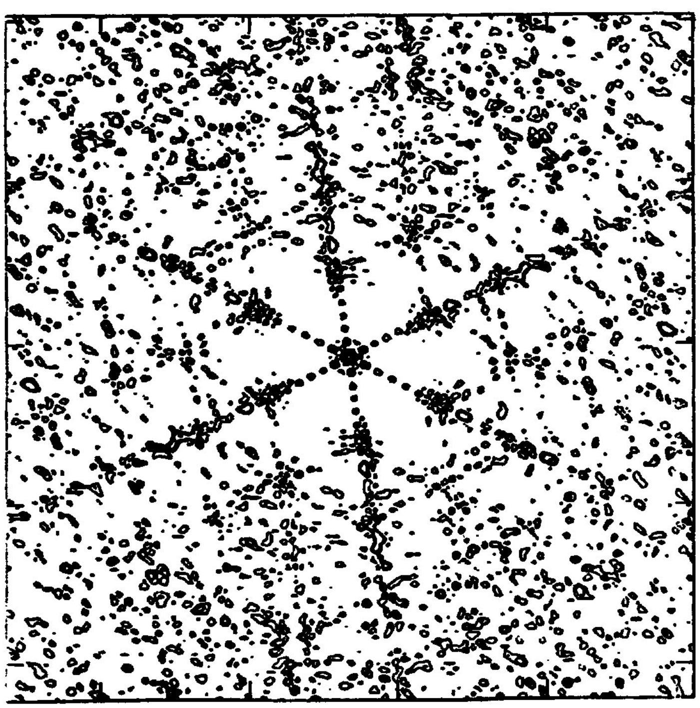

Figure 3(a)
Super-Uniform Beam
Averaging Neighbourhood = 3x3

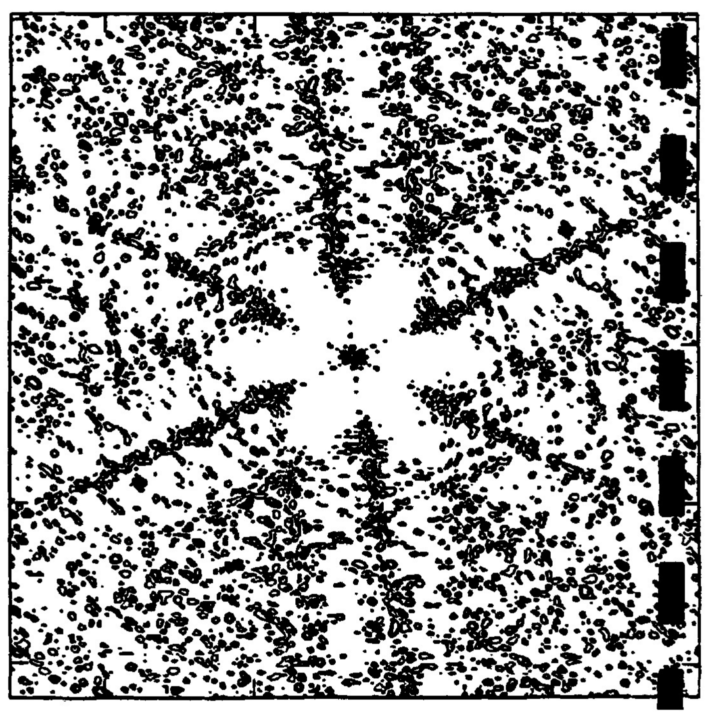

Figure 3(b)
Super-Uniform Beam
Averaging Neighbourhood = 7x7

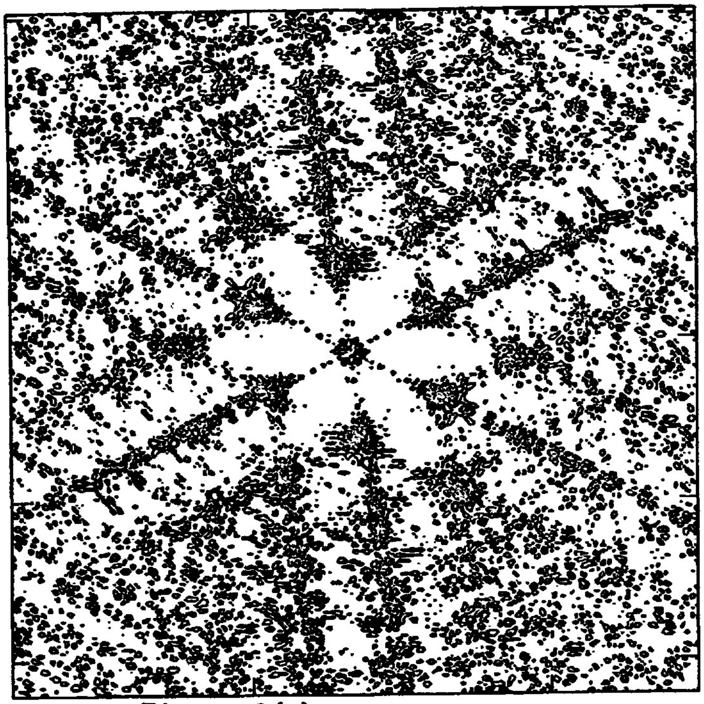

Figure 3(c)
Super-Uniform Beam
Averaging Neighbourhood = 11x11

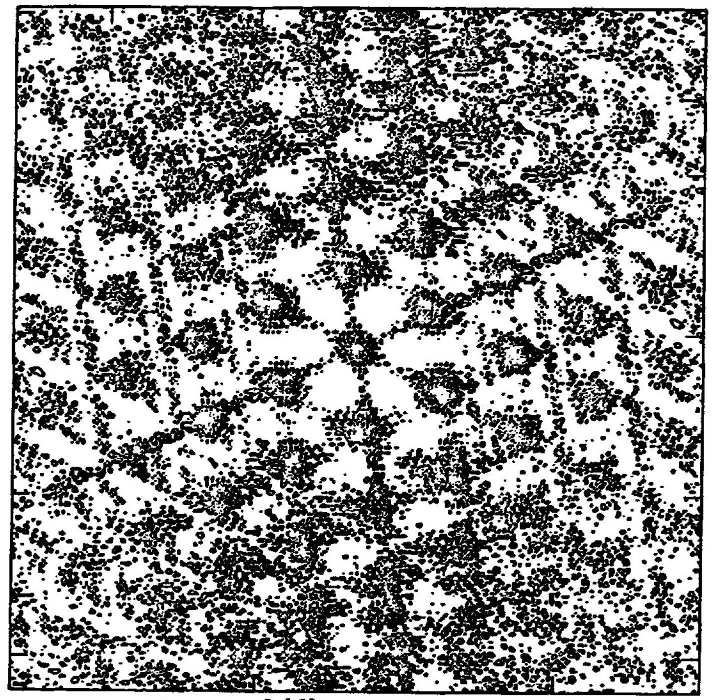

Figure 3(d)
Super-Uniform Beam
Averaging Neighbourhood = 15x15

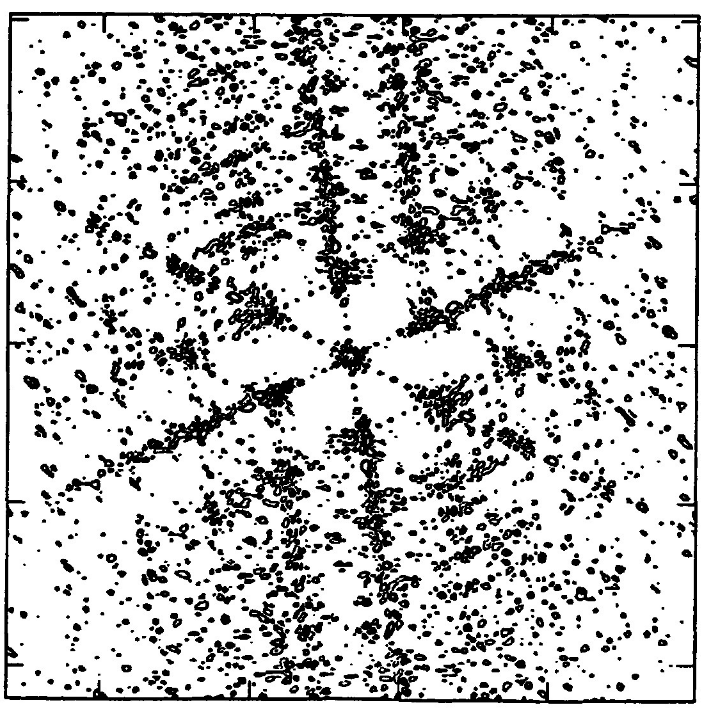

Figure 4
Optimal Weighted Beam from
Sinc Lobe Window Function
Roughly equivalent to Figure 3(c)

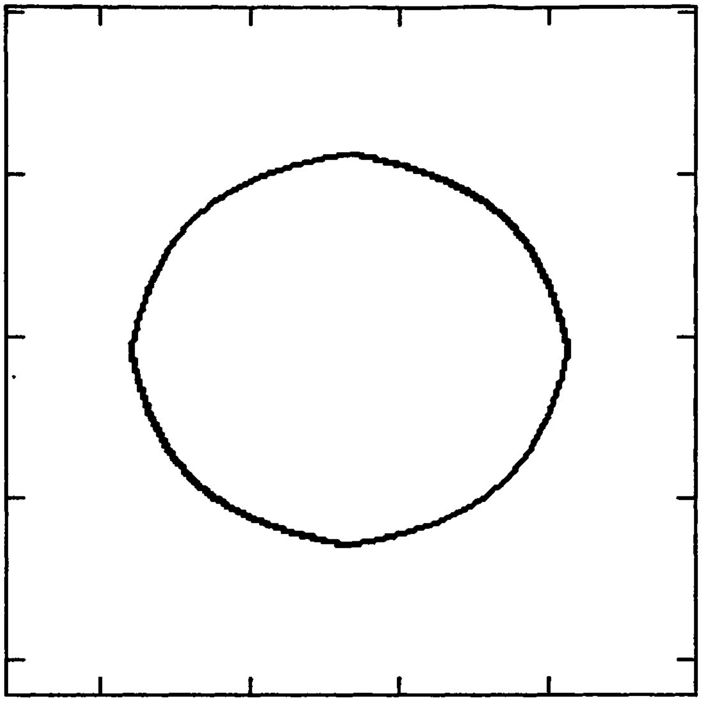

Figure 5(a)
Large Disc Window Function

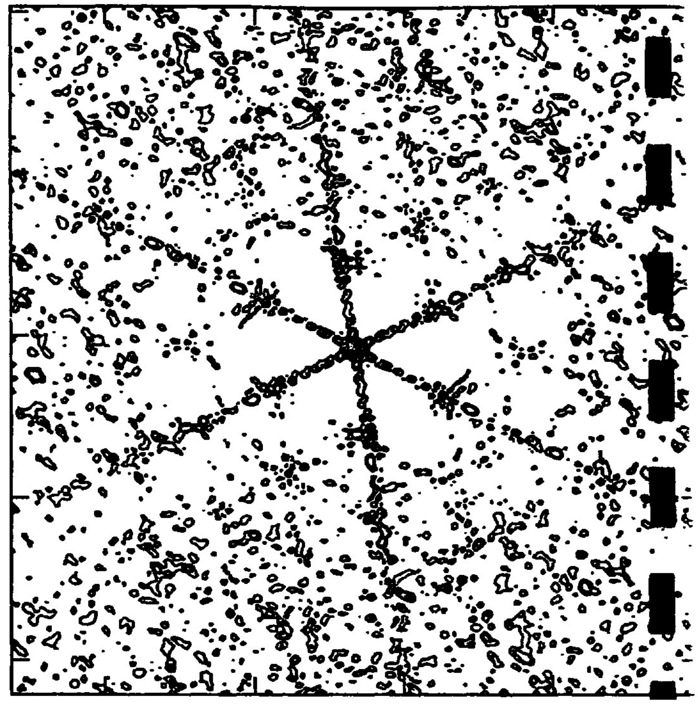

Figure 5(b)
Optimally Weighted Beam

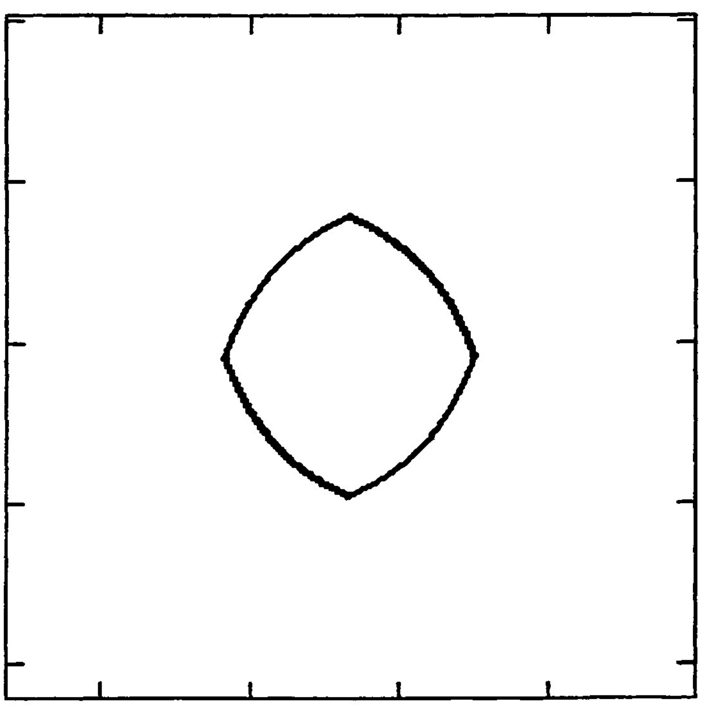

Figure 6(a)
Medium Disc Window Function

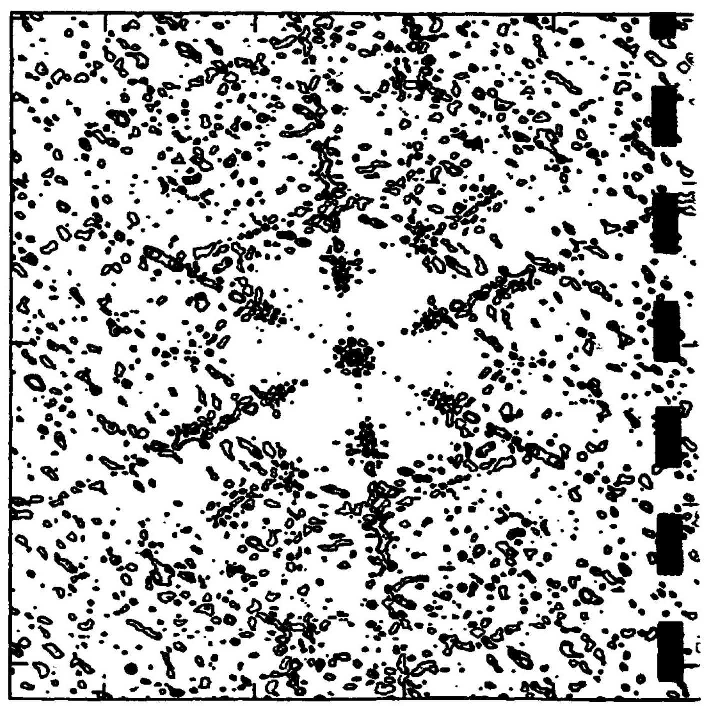

Figure 6(b)
Optimally Weighted Beam

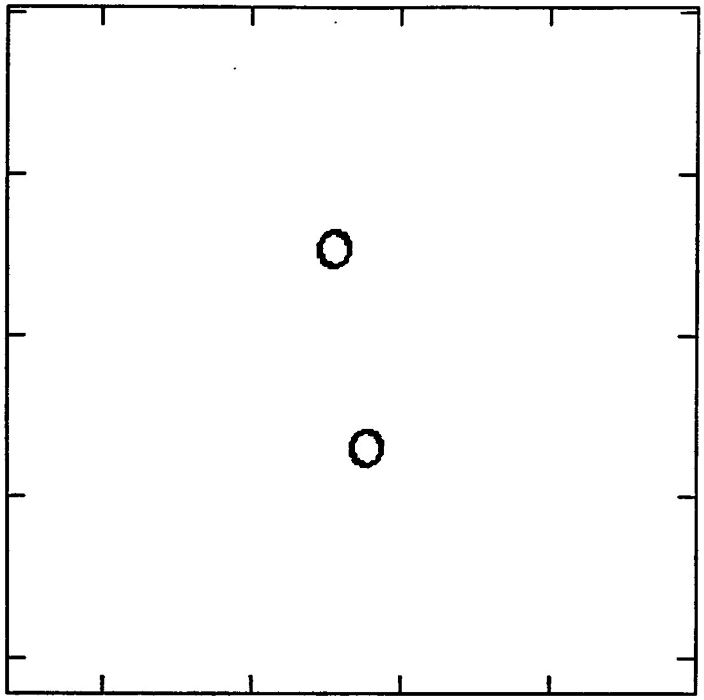

Figure 7(a)
Two Small Disc Window Function

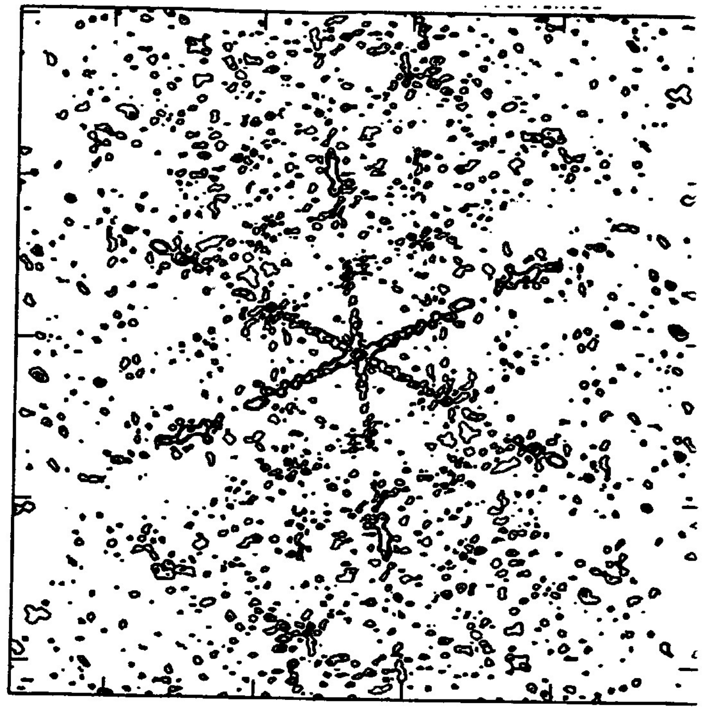

Figure 7(b)
Optimally Weighted Beam

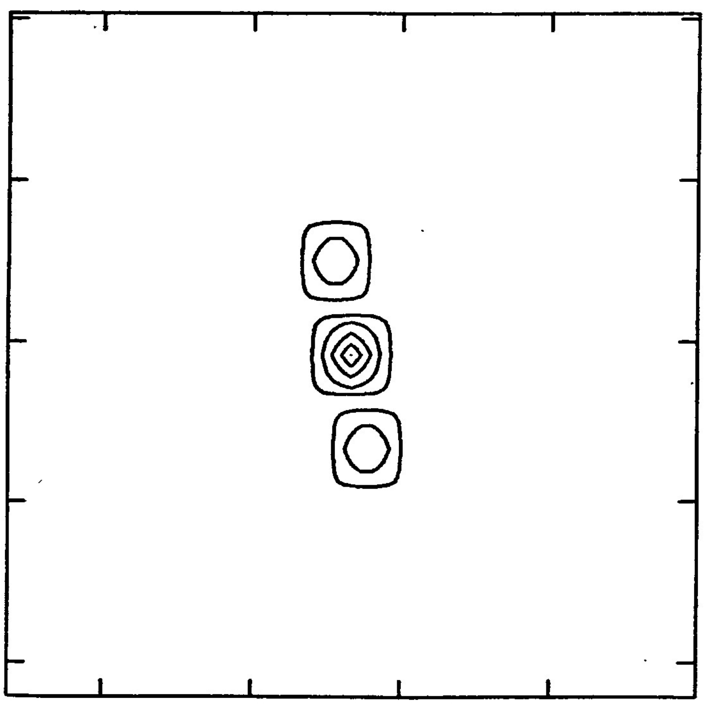

Figure 8(a)
Two Box Autocorrelation Window

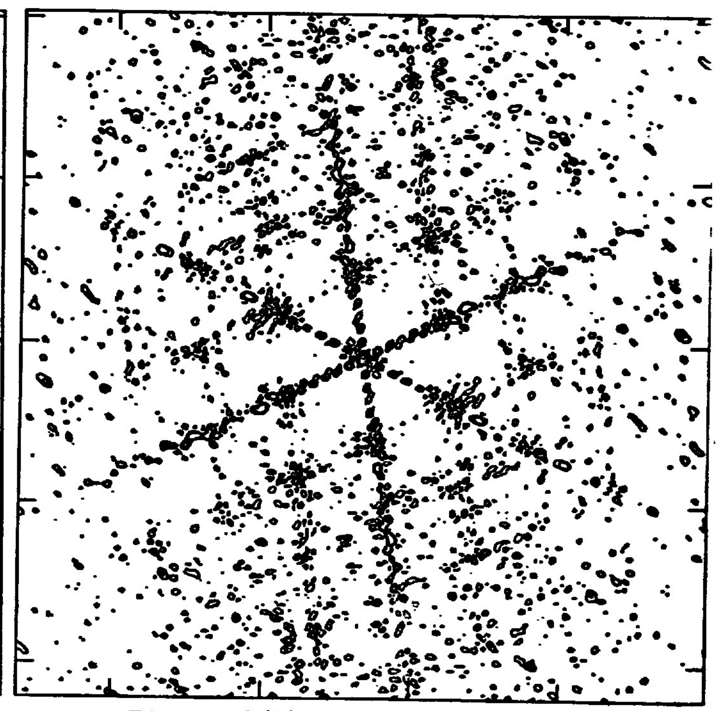

Figure 8(b)
Optimally Weighted Beam