Automation of Imaging Simulations
for Array Configurations using Classic AIPS

Steven Heddle†and Adrian Webster‡
†UK Astronomy Technology Centre/‡Institute for Astronomy
Royal Obervatory Edinburgh
Blackford Hill
Edinburgh EH9 3HJ, UK
(sbh,asw)@roe.ac.uk
6th March 2000

Abstract

Since the ALMA telescope will consist of about 60 antennas that may be deployed on 200 - 300 stations, the number of configurations to be modelled is large, particularly if the array will be operated as a zoom telescope in which successive observing configurations differ only in the position of one antenna. Since, moreover, the electromagnetic performance has to be modelled in snapshot and earth-rotation modes, and at a number of different declinations, modelling even a representative selection of the possible overall configurations constitutes an onerous and time-consuming task if it is not heavily automated. This Memo describes some progress that has been made towards this end using straightforward shell scripts applied while running Linux, with Classic AIPS as the core simulation tool. The scripts generate AIPS batch files for imaging simulations automatically, by inputting key parameters on the command line. At present these are declination, wavelength, length of track, and filename of antenna positions. The results presented include uv coverage, distribution of baselines, dirty beams and maps, and slices through the dirty beam. The method and scripts are described here, and results from a hybrid composite zoom design are presented. The method allows standard, comparable sets of results to be obtained quickly and easily.

1 Introduction and overview

This work was driven by a need to simulate the imaging properties of zoom arrays at the various zoom settings, and the need to produce comparable sets of results. In the zoom array considered, 60 antennas were placed on consecutive stations from a total set of 240, numbered by increasing distance from the array centre. Thus the most compact configuration occupies stations 1 to 60, the highest resolution configuration occupies stations 181 to 240, with a further 179 configurations in between. Even simulating every 9th configuration leads to 21 configurations to be evaluated at various declinations (e.g. 0, -23 and -60 degrees) and for both snapshots and long tracks. Clearly it would be inefficient to perform such simulations interactively (at an AIPS session, say), and also the opportunity to introduce inconsistencies would be huge.

Using Classic AIPS (the 15APR99 version, with bug fixes in UVCON and UVSIM) as the core simulation tool we have succeeded in automating the processing of sets of array configurations (and indeed single configurations) using simple Unix shell scripts. The basic method is outlined next and expanded upon in the following sections.

2 Basic approach

We first defined the results we wished to obtain for each and all the configurations. A prototype AIPS batch job was written which achieved this for a given configuration. Then by defining a systematic naming convention for the configuration, it was possible to embody the properties of the configuration and the nature of the simulation in the names of the AIPS data files, both in the internal catalog and in the exported results. More importantly this allowed a script to be written which used the rules of the naming convention to write the appropriate batch file automatically, which then generated the same type of results as for the prototype, in a consistent manner. This script could then be called repeatedly from a 'master' script which finds the data files for the antenna positions for each configuration and reads the necessary command line arguments, outputting a single batch file to simulate all configurations in the same manner. AIPS is then invoked with the batch file as standard input, and all being well, the results follow.

The results obtained show uv coverage, distribution of baselines, dirty maps and beams, and slices across the dirty beam. The results may be obtained for various declinations and wavelengths as specified on the master script's command line, and for snapshot and 6 hour tracks. It is intended to encompass CLEANING and/or MEM when a consensus on the appropriate parameters is reached, and also to allow specification of different test images for each array configuration- at present an image of M51 (as available from John Conway's website– http://www.oso.chalmers.se/~jconway/ALMA/IMAGES/) is hardwired into the processing.

3 Formatting of antenna position data

For the zoom array under consideration, the coordinates of all 240 array stations relative to the array centre were provided as a simple ascii file, with one pair of space separated coordinates per line, ordered by increasing distance from the array centre. The individual configurations are placed on 60 consecutive stations, and may be identified by the index of the first station. Having extracted 60 sets of coordinates for a configuration, they must then be incorporated in a file formatted to suit the UVCON task. Scripts were written to extract a specified number of consecutive coordinates from a list given a start point (antSelect.csh), and to to format a list of coordinates for UVCON (uvconFormat.csh)– see Appendix A.

The correctly formatted files were given names with a 5 character root and arbitrary extension. The root is subsequently used to generate the names of the data files, so here it is sensible to make the name a composite of the starting array station and complete it with 2 characters which specify the array type (e.g. 1A1.uvc, 181A1.uvc etc.)

The positioning of the array stations is shown in Fig 1, with some example configurations in Figure 2. Their naming convention will be explained in the next section.

4 Naming convention for configurations

It was apparent that using a naming convention for the array configurations and the resultant data files would simplify the construction of the scripts to generate the batch files, and also aid the understanding of the data if the name could embody the main parameters of the simulation. The file names would have to also be less than 12 characters to avoid being truncated by AIPS. The convention used is as follows, and embodies the array configuration, length of track, declination and wavelength...

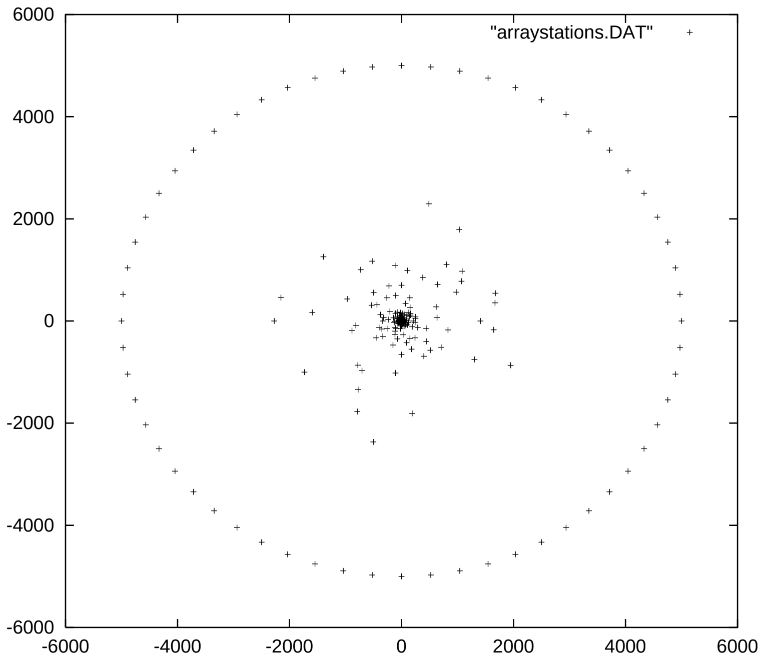
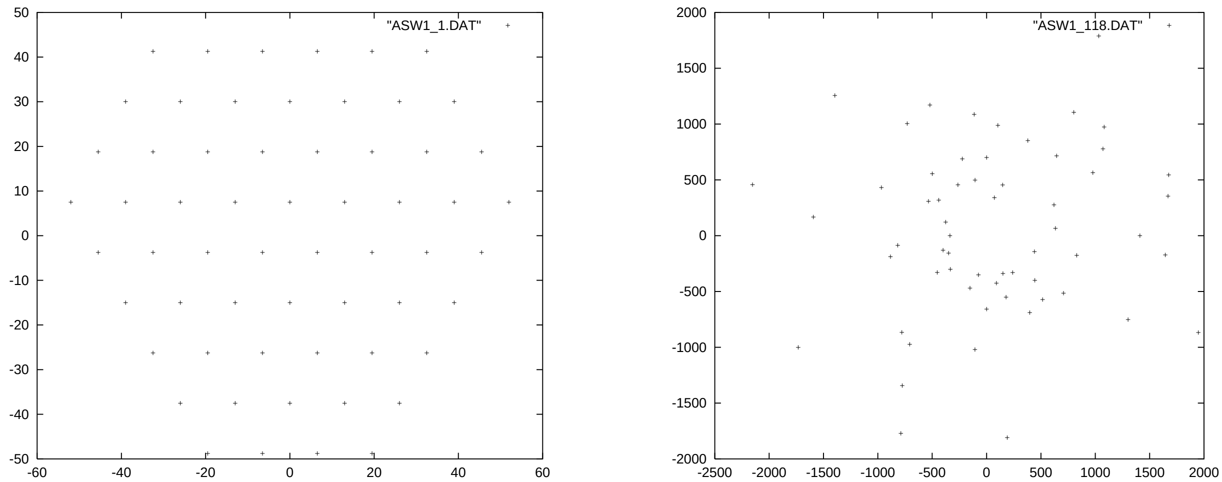


Figure 1: Full set of antenna stations (axes in m).



a)1A1 configuration

b)118A1 configuration

Figure 2: Example configurations (axes in m).

1. **Array configuration-** the current set of stations leads to A1 type arrays, prefixed by the starting station, giving 1A1, 10A1...181A1. Max 5 characters.
2. **Length of track-** zenith snapshots are SN, 6 hour tracks are _6, 4 hour tracks are _4 etc. This assumes that the tracks we will be looking at here are symmetrical about zenith. Other tracks will be labelled according to some alphabetic convention, to be defined. Max 2 characters.
3. **Declination-** is designated by a signed integer, e.g. +0, -23, -30, -60. Max 3 characters.
4. **Wavelength-** is designated by an underscore followed by the approximate wavelength as an integer no. of mm, e.g. _1, _3. Max 2 characters.

So for a set of results at 3mm, the name is constructed as follows:

Config	track length	declination	filename
1A1	SN	0	1A1SN+0_3
		-23	1A1SN-23_3
		-30	1A1SN-30_3
		-60	1A1SN-60_3
	6 hours	0	1A1_6+0_3
		-23	1A1_6-23_3
		-30	1A1_6-30_3
		-60	1A1_6-60_3

Table 1: Construction of filenames

5 Data to be obtained from prototype simulation

For this experiment we determined a set of results to be obtained for each simulation. This set is of course neither definitive nor set in stone.

Results to be obtained for each simulation are:

- a plot of the uv coverage of the array configuration
- a histogram of the baseline lengths of the array configuration
- a listing of the uv data for the configuration to be exported from aips
- a contour map of the dirty beam hard clipped at 10% of the peak intensity, and at the same scale as the reference image (in this case M51)
- a dirty map of M51 obtained using the antenna configuration, at the same scale as the reference image
- line graphs of slices though the dirty beam at angles of 0, 45, 90 and 135 degrees respectively, showing the dirty beam up to 100% of the peak value.

N.B. This data is calculated for snapshots and 6 hour tracks for each combination of array, wavelength, and declination specified.

The listing of the uv data is not practicable for the long tracks, as only a limited number of points may be exported using the appropriate AIPS task.

6 Creating batch files for simulations

Having defined the namimg convention and specified the data to obtain, we can write the scripts which will automate the simulation process. The first script generates a batch file which contains the AIPS commands necessary to carry out the simulations for a given array and write the data to appropriately named files. The naming of files is determined by the script's command line arguments which are array_name, declination, and wavelength, and also an argument to specify which AIPS disk to use for the internal catalog entries (usually set to 0). Appropriate values of the AIPS adverbs are saved in a file which is retrieved using GET to minimise definition and change of adverbs in the batch file. This script, batCreate.csh, is documented more fully in its header, and is listed as Appendix B.

The script actually run by the user is processConfigs.csh, included here as Appendix C. This script takes the similar command line arguments to batCreate.csh, except that arrayname is replaced by the name of the file which contains the set of filenames of all the arrays to be processed, and that several declinations may be specified. An additional argument specifies the name of the output batch file. processConfigs.csh repeatedly calls batCreate.csh to cover the permutaions of array, declination and wavelength, and adds essential job control commands to be passed to AIPS when invoked for the batch job. An excerpt of the resulting file output file is included here as Appendix D.

This effort is justified when it is considered that doing 21 configurations at 4 declinations interactively would take approximately 20 tedious working days.

7 What actually has to be done

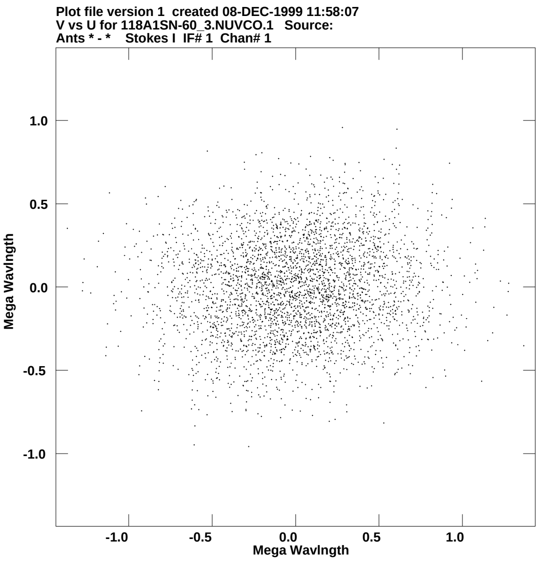
We start with the file of antenna station positions and use antSelect.csh to create a series of files containing the antenna positions for each configuration to be simulated. These files in turn are formatted for subsequent use by UVCON using uvconFormat.csh, and at this stage we name the files so that the root of the filename conforms to the naming convention described earlier (the tail of the filename is not important) e.g. 1A1.UVC...181A1.UVC. The filenames of the configurations which we wish to simulate are then collected into a text file with one filename per line, and the name of this text file is supplied to processConfigs.csh as an argument, along with the name of the output batch file, the wavelength in mm, the AIPS disk to use (generally set to 0 = any available disk) and the declinations. processConfigs uses this information to repeatedly call batCreate.csh to generate the output batch file which contains the commands to perform the simulations for snapshots and 6 hour tracks for the configurations, wavelength and declinations specified, and export the results as sensibly named (i.e. according to the same convention) files, with the tails distinguishing the type of result. The batch file is executed by simply directing it as input to an AIPS session invoked from the Unix shell prompt as follows:

aips ¡ batch_file

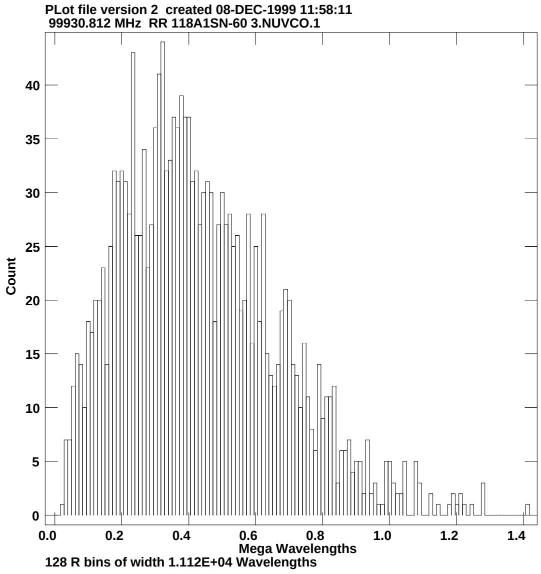
The AIPS password is prompted for, and everything in the batch file is executed automatically thereafter.

8 Typical results obtained

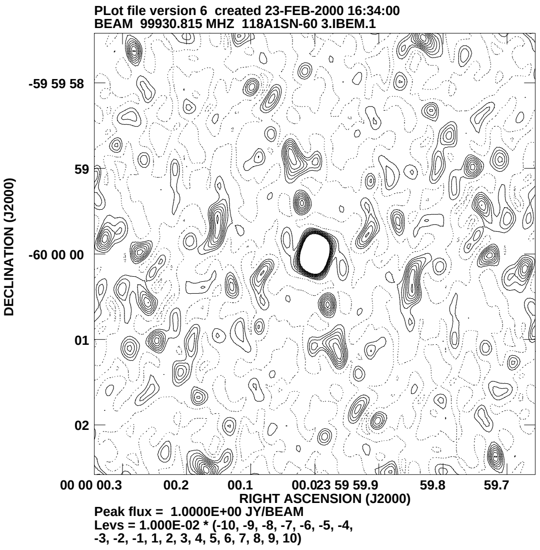
The data set for which results are presented is 118A1SN-60_3, i.e. the index of the first antenna station is 118, as part of the A1 configuration, and it is a snapshot at a declination of -60 degrees and a wavelength of 3mm. Only the figures are presented for this dataset, with the textual uv data file omitted. These results are shown in Figure 3, a) – h).



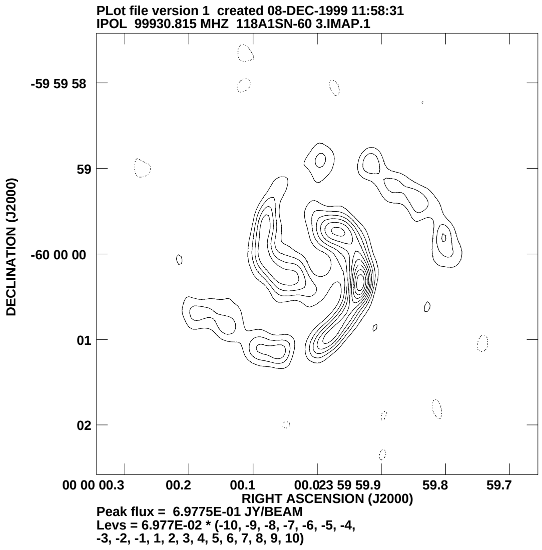
a) uv coverage



b) distribution of baselines

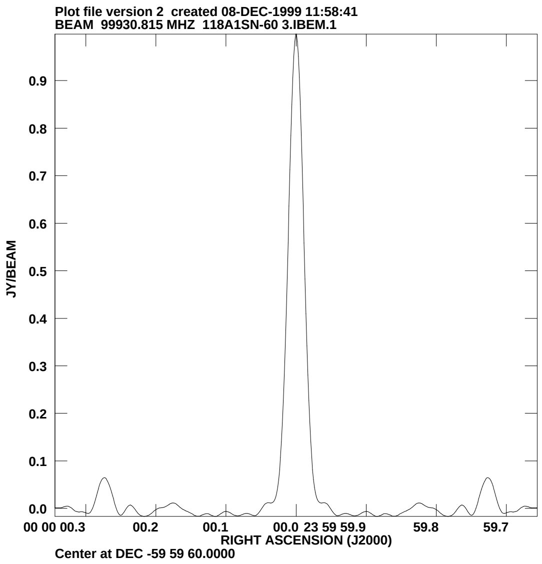


c) dirty beam

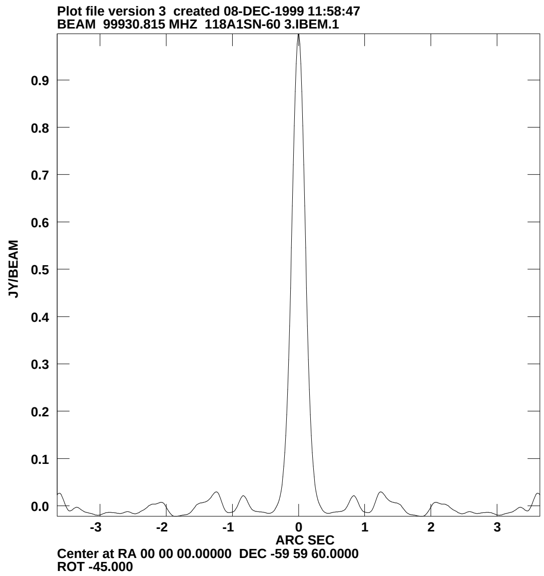


d) dirty map

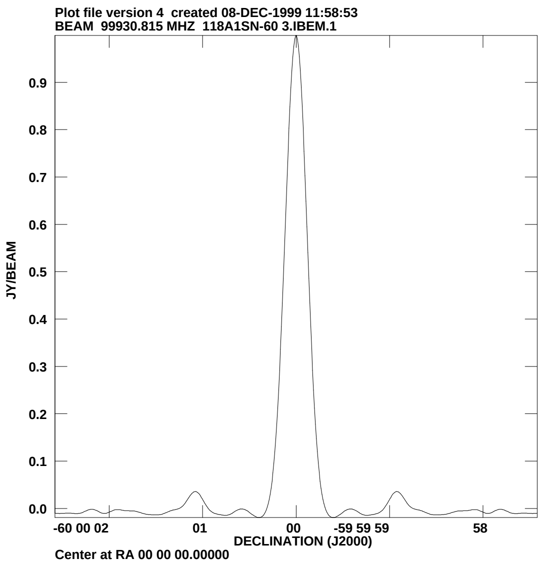
Figure 3: Typical data from simulations, a)–d)



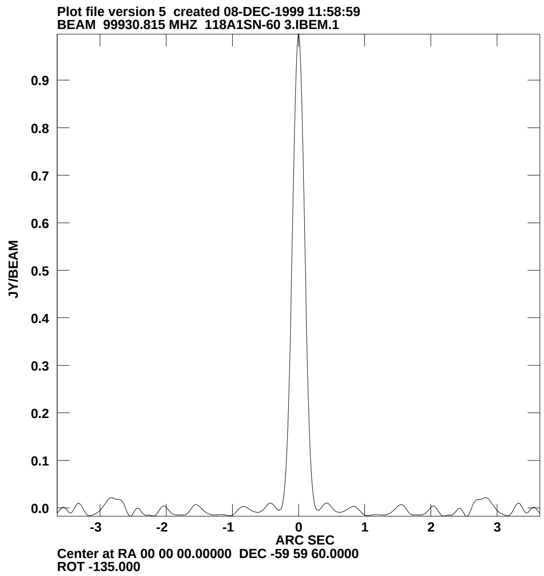
e) slice through dirty beam at 0 degrees



f) slice at 45 degrees



g) slice at 90 degrees



h) slice at 135 degrees

Figure 3: Typical data from simulations, e)-h)

9 Conclusions

The process outlined goes some way to providing an automated atandardised technique for analysing array configurations. The time saving is large, particularly in the case of zoom arrays- 21 antenna configurations of the kind described, at 3 declinations and for both snaphots and 6 hour tracks took less than four hours to simulate, generating 464MB of results in the form of uv data and PostScript formatted graphs. The simulations at present do not encompass CLEAN or MEM techniques. However, if standard non-interactive procedures for these techinques can be defined, the automated simulations should be certainly be able to be extended to include them. Also the imaging proceeds using only a single image of M51 for test and reference– clearly it would be desirable to extend the process to include a range of images applicable to the expected imaging characteristics of each specific array configuration. Again, this should be possible. The process may as easily be applied to a single configuration as a range of configurations.

We present these results as a preliminary example of a possible standard analysis procedure for candidate array configurations.

Known bug

When the aips disk in use fills up during execution of the batch file (i.e.while a catalog entry is being written) the aips session may die. In that case the point reached should be noted, and the batch commands already executed deleted from the batch file EXCEPT FOR THE FIRST SIX which set up the batch job. The edited batch file can be resubmitted, and everything should churn away happily again. A better fix than this will be found!

Appendix A: Scripts to extract and format coordinates for UVCON

antSelect.csh

```
#! /bin/csh

# File:    antSelect.csh
# Author:  Steven Heddle, heddle@cyberdude.com
# Date:    18th November 1999
# Version: 1.0

# Script to simplify select of antenna coordinates from a file of candidate
# positions.
#
# N.B. Assumptions:
# =================
# 1) Antennas are numbered from 1, according to their line numbers in the
#    file of candidate positions
# 2) The file of candidate positions has coordinates for each antenna on
#    separate consecutive lines.
# 3) Selected antennas are to form a consecutive block of sets of
#    coordinates from the allowed positions.


# Invocation:
# antSelect.csh file_of_station_coords index_of_first_antenna
                             number_of_antennas > outfile

if (${#argv} != 3) then
    echo "Usage: antSelect.csh <file of station coords> <index of first
antenna>
                           <number of antennas>"
    exit
endif

@ FINISH = ($2 + $3 -  1)

if ($2 == 0) then
echo "Indexing of antenna positions starts at 1"
exit
endif

if ($FINISH > `cat $1 | wc -l`) then
echo "Not enough stations to allow this, sorry"
exit
endif


cat $1 | head -$FINISH | tail -$3
```

uvconFormat.csh

```
#! /bin/csh

# File:    uvconFormat.csh
# Author:  Steven Heddle, heddle@cyberdude.com
# Date:    10th November 1999
# Version: 1.0

# The purpose of this script is to provide a quick and simple way of
# converting a list of 2 or 3 dimensional coordinates for antenna positions
# to a file correctly formatted for use by UVCON in AIPS (corresponding to
# the INFILE adverb). The input list is assumed to provide 2 (x,y) or 3
(x,y,z)
# coordinates for the position of each antenna, on separate consecutive
lines.
#
# Invocation:
# uvconFormat.csh infile > outfile
#
# The variables set below correspond to sensible defaults for ALMA
# array modelling, but provide a convenient means of changing parameters
# such as antenna size and temperature without having to edit most of the
# lines of an existing UVCON input file.
#
# The meaning of the variables are self explanatory with reference to the
# following description (from AIPS) of the UVCON INFILE format:

#                SPECIFYING THE ARRAY CONFIGURATION
#
#     The information defining the array configuration and
# antenna characteristics is read by
# UVCON from an auxiliary input file, supplied by the user.  This
# is a free-format text file.  One must list, in the following
# order:
#
# Line 1: The number of antennas,
# Line 2: The site latitude, in degrees (so that elevation angles
# may be computed), and the site longitude, in degrees, positive
# to west (so that the antennas location given in LAT, LONG
# coordinates can be recalculated to the celestial coordinate system.
# Line 3: A multiplicative conversion factor specifying how the
#         antenna coordinates, listed next by the user, can be
#         converted into units of meters; and a second
#         multiplicative conversion factor specifying how the listed
#         antenna diameters can be converted into units of meters.
#     If the antenna location coordinates are given in nanoseconds,
# the conversion factor is 0.299.
```

```
#
# The remaining lines specify the antenna location and paramters,
# with one line for each antenna.  Each line is formatted thus:
#
# Col. 1: The coordinate system:
#    0 => Earth Equatorial, with X positive towards HA=0, Dec=0,
# Y positive towards HA=-6, Dec = 0, Z positive to NCP.
# Units in meters, but see Line 3 description above.
# Coordinate origin is at the array center.
#                 Warning: VLBA uses opposite dirrection for Y axis.
#                          So you need to change it!
#    1 => Local Horizon, with X positive towards east,
# Y positive towards north, Z positive to local zenith.
# Units in meters, but see Line 3 description above.
# Coordinate origin is at the array center.
#    2 => Geodetic, with coordinates given by latitude,
# longitude (positive towards west), (both in degrees)
# and elevation above sea level (in meters).
#
# Col. 2: Antenna Coordinate X, as defined above.
# Col. 3: Antenna Coordinate Y, as defined above.
# Col. 4: Antenna Coordiante Z, as defined above.
# Col. 5: Antenna diameter.
# Col. 6: Antenna efficiency (fraction). 0 => 0.5
# Col. 7: Antenna system temperature (K) 0 => 50K
# Col. 8: Number of levels of digitization of signals. 0 => 2 level
# Col. 9: 0 => the array center is at the given center (Line 2) (default)
#         1 => the array center is at the Earth center (VLBA case)


set LATITUDE=-23.0
set LONGITUDE=67.6999969
set COORDSCALE=1
set ANTDIAMSCALE=1
set COORDSYS=1
set ZCOORD=0
set ANTDIAM=12
set ANTEFF=0.5
set ANTSYSTEMP=50
set DIGLEVEL=1
set CENTRETYPE=0

if (${#argv} != 1) then
    echo "Usage: uvconFormat.csh <file of coordinates>"
    exit
endif

# check number of columns

set NCOLS=`head -4 $1 | tail -1 | awk '{print NF}'`
```

```
if($NCOLS != 2 && $NCOLS != 3) then
    echo "Wrong number of fields for this script (not 2 or 3)"
    exit
endif

echo `cat $1 | wc -l`                   # line 1
echo $LATITUDE $LONGITUDE               # line 2
echo $COORDSCALE $ANTDIAMSCALE          # line 3

# remaining lines of 9 columns of data per antenna...

if($NCOLS == 2) then
cat $1 | awk -v f1=$COORDSYS -v f4=$ZCOORD -v f5=$ANTDIAM -v f6=$ANTEFF -v
        f7=$ANTSYSTEMP -v f8=$DIGLEVEL -v f9=$CENTRETYPE
        '{printf ("%s %f %f %s %s %s %s %s %s\n",f1, $1,
$2,f4,f5,f6,f7,f8,f9)}'
else
cat $1 | awk -v f1=$COORDSYS -v f5=$ANTDIAM -v f6=$ANTEFF -v
        f7=$ANTSYSTEMP -v f8=$DIGLEVEL -v f9=$CENTRETYPE
        '{printf ("%s %f %f %s %s %s %s %s %s\n",f1, $1,
$2,f4,f5,f6,f7,f8,f9)}'
endif
```

Appendix B– batCreate.csh

```
#! /bin/csh
echo
# File:    batCreate.csh
# Author:  Steven Heddle, heddle@cyberdude.com
# Date:    1st December 1999
# Version: 1.2

# Version history:
# V1.1 corrects misinterpretation of the PLEV parameter- should be set at
# 1 to give +-10% display of IBEM, and error in setting INDISK
# V1.2 Plev 0 added before plotting IMAP to get correct contour range

# Invocation:
# batCreate.csh array_name declination wavelength disk

# Script to write a batch file for processing by the AIPS standalone task
BATER
# using candidate antenna arrays for the ALMA.
# The data in this script must be preceded by the BATER commands BATQUE 1,
# BATCLEAR and BATCH and followed by the commands ENDBATCH and SUBMIT. These
# commands will commonly be provided by a further script, embedded in which
# will be successive calls to batCreate.csh. When the batch file created by
# this script is executed, aformmated array of antenna positions is read,
and
# the uv coverage calculated. Visibilities are also calculated for an input
# model of M51. Up to 2000 sampled points of the uv data are exported to a
# text file in the directory specified by the MYAREA environmental variable,

# as are a number of plots stored as PostScript files. These are
# i)    the uv plane, stored as *.UV.PS
# ii)   a histogram of the radial uv distances
# iii)  a contour plot of the dirty image, *.IMAP.PS
# iv)   a contour plot of the dirty beam below the 10% level, *.IBEM.PS
# v)    a slice through the centre of the dirty beam at 0 degrees, *.SL0.PS
# vi)   a slice through the centre of the dirty beam at 45 degrees,
*.SL45.PS
# vii)  a slice through the centre of the dirty beam at 90 degrees,
*.SL90.PS
# viii) a slice through the centre of the dirty beam at 135 degrees,
*.SL135.PS
# The * denotes a filename made up as follows:
# <root of array name><SN or _6 depending on whether snapshot or 6hr track>
# <signed declination, (positive decs have the plus added by the script)>
# _<wavelength in mm>
# e.g 181A1SN-23_3, for a snapshot of the 181A1 array at 3mm and a dec of
-23
#
# The input file of array positions is read from the directory specified by
```

```
the
# environmental variable $ASWDATA
#
# The set of adverbs stored as uvcon3mm must be correct and available.
# Different hour angles for the tracking may easily be set by reference to
the
# comments below

if (${#argv} != 4) then
    echo "Usage: batCreate.csh <arrayname> <declination>
                                       <wavelength> <disk[1-4]>"
    exit
endif

set ARRAY=$1
set DEC=$2
if ($2 >= 0) then
   set DEC=+$2
endif
set WAVELENGTH=$3
@ FREQUENCY = ( 299793 / $3 )
set DISK=$4
##################for snapshot

## set adverbs for NUVCO and run it
echo "get nuvco3mm"                    # get predefined adverbs
echo "indisk "$DISK
echo "outdisk "$DISK
echo "aparm(1)=0"                      # ignore set frequency
echo "aparm(2)=0."$WAVELENGTH          # set wavelength (in cm)
echo "aparm(3)="$DEC                   # set declination
echo "aparm(4)=0"                      # set min hour angle
echo "aparm(5)=0"                      # set max hour angle
echo "aparm(7)=180"                    # set integration time(s)
echo "infile 'aswdata:"$ARRAY"'"       # set array
echo "outname '"$ARRAY:r"SN"$DEC"_"$WAVELENGTH"'"     # rename output file
echo "go nuvco"                        # calculate uv coverage, visibilities

                                       # for input image of M51

## create plot file for uv plane
echo "inname '"$ARRAY:r"SN"$DEC"_"$WAVELENGTH"'" # use newly created input
echo "inclass 'nuvco'"                           # ...of class nuvco
echo "inseq 0"                                   # ...and most recent
echo "bparm 6,7,2,0"                             # plot u against v
echo "dotv -1"                                   # to be written to plot
file
echo "go uvplt"                                  # plot

echo "bparm 0"                                   # reset shared bparm for
```

```
LWPLA
echo "bparm(6)=4"                                # use A4 paper size
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".UV.PS'" #name of PS
file
echo "go lwpla"                                  # write plot to PS file

## create histogram of uv distance
echo "nboxes 128"                                # set number of bins

echo "axtype 'R'"                                # plot radial distance
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".HIST.PS'" # PS file
echo "go uvhgm"                                  # create hist as plot file

echo "go lwpla"                                  # write plot to PS file
echo "nboxes 0"                                  # reset nboxes for later
tasks

## export uv data to text file
echo "infile '"$ARRAY:r"SN"$DEC"_"$WAVELENGTH"'" # use newly created input
echo "inclass 'nuvco'"                           # ...of class nuvco
echo "inseq 0"                                   # ...and most recent
echo "docrt 0"                                   # output to go to a file
echo "xinc 1"                                    # output every xinc'th pt
echo "ncount 2000"                               # max number of points
echo "outprint 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".TXT'" # PS file
echo "go prtuv"                                  # print uv data to file

## prepare and execute imaging task
echo "bchan 0"
echo "outname ' '"
echo "outdisk "$DISK
echo "in2name ' '"
echo "in2class ' '"
echo "in2seq 0"
echo "in2disk "$DISK
echo "selfreq "$FREQUENCY
echo "go imagr"
echo "in2disk 0"

## plot dirty map at same scale as input image of M51
echo "inclass 'imap'"                  # class of newly created image

echo "inseq 0"                         # ...using most recent of that name
echo "blc 426,427"                     # bottom left point of plotted area
echo "trc 598,599"                     # top right point of plotted area
echo "plev 0"
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".IMAP.PS'" # PS file
echo "go cntr"                         # do contour plot of defined area
echo "go lwpla"                        # write plot to PS file
```

```
## plot dirty beam up to 10 percent of maximum flux
echo "inclass 'ibem'"                  # class of newly created image
echo "plev 1"                          # set perecentage level
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".IBEM.PS'" #PS file
echo "go cntr"                         # do contour plot of defined area
echo "go lwpla"                        # write plot to PS file

## plot slice through dirty beam at 0 degrees
echo "blc 426,513"                     # start point of slice
echo "trc 598,513"                     # end point of slice
echo "plev 0"                          # set percentage level
echo "outfile ' '"                     # don't write to a file, plot
echo "go slice"                        # calculate slice
echo "go sl2pl"                        # turn slice into a plot file
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".SL0.PS'" # PS file
echo "go lwpla"                        # write plot to PS file

## plot slice through dirty beam at 45 degrees
echo "blc 426,427"
echo "trc 598,599"
echo "plev 0"
echo "outfile ' '"
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1sn-23_3.sl45.ps'"
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".SL45.PS'"
echo "go lwpla"

## plot slice through dirty beam at 90 degrees
echo "blc 512,427"
echo "trc 512,599"
echo "plev 0"
echo "outfile ' '"
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1sn-23_3.sl90.ps'"
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".SL90.PS'"
echo "go lwpla"

## plot slice through dirty beam at 135 degrees
echo "blc 426,599"
echo "trc 598,427"
echo "plev 0"
echo "outfile ' '"
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1sn-23_3.sl135.ps'"
echo "outfile 'myarea:"$ARRAY:r"SN"$DEC"_"$WAVELENGTH".SL135.PS'"
echo "go lwpla"
```

```
##################for 6 hour track, from -3 to +3

## set adverbs for NUVCO and run it
echo "get nuvco3mm"                    # get predefined adverbs
echo "aparm(1)=0"                      # ignore set frequency
echo "aparm(2)=0."$WAVELENGTH          # set wavelength (in cm)
echo "aparm(3)="$DEC                   # set declination
echo "aparm(4)=-3"                     # set min hour angle
echo "aparm(5)=3"                      # set max hour angle
echo "aparm(7)=180"                    # set integration time in seconds
echo "indisk "$DISK
echo "outdisk "$DISK
echo "infile 'aswdata:"$ARRAY"'"       # set array
echo "outname '"$ARRAY:r"_6"$DEC"_"$WAVELENGTH"'"     # rename output file
echo "go nuvco"                        # calculate uv coverage, visibilities

                                       # for input image of M51

## create plot file for uv plane
echo "inname '"$ARRAY:r"_6"$DEC"_"$WAVELENGTH"'" # use newly created input
echo "inclass 'nuvco'"                           # ...of class nuvco
echo "inseq 0"                                   # ...and most recent
echo "bparm 6,7,2,0"                             # plot u against v
echo "dotv -1"                                   # to be written to plot
file
echo "go uvplt"                                  # plot

echo "bparm 0"                                   # reset shared bparm for
LWPLA
echo "bparm(6)=4"                                # use A4 paper size
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".UV.PS'" #name of PS
file
echo "go lwpla"                                  # write plot to PS file

## create histogram of uv distance
echo "nboxes 128"                                # set number of bins

echo "axtype 'R'"                                # plot radial distance
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".HIST.PS'" # PS file
echo "go uvhgm"                                  # create hist as plot file

echo "go lwpla"                                  # write plot to PS file
echo "nboxes 0"                                  # reset nboxes for later
tasks

## export uv data to text file
#echo "infile '"$ARRAY:r"_6"$DEC"_"$WAVELENGTH"'" # use newly created input
#echo "inclass 'nuvco'"                           # ...of class nuvco
#echo "inseq 0"                                   # ...and most recent
#echo "docrt 0"                                   # output to go to a file
```

```
#echo "xinc 1"                                   # output every xinc'th pt
#echo "ncount 2000"                              # max number of points
#echo "outprint 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".TXT'" # PS file
#echo "go prtuv"                                 # print uv data to file

## prepare and execute imaging task
echo "bchan 0"
echo "outname ' '"
echo "outdisk "$DISK
echo "in2name ' '"
echo "in2class ' '"
echo "in2seq 0"
echo "in2disk "$DISK
echo "selfreq "$FREQUENCY
echo "go imagr"
echo "in2disk 0"

## plot dirty map at same scale as input image of M51
echo "inclass 'imap'"                 # class of newly created image

echo "inseq 0"                        # ...using most recent of that name
echo "blc 426,427"                    # bottom left point of plotted area
echo "trc 598,599"                    # top right point of plotted area
echo "plev 0"
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".IMAP.PS'" # PS file
echo "go cntr"                        # do contour plot of defined area
echo "go lwpla"                       # write plot to PS file

## plot dirty beam up to 10 percent of maximum flux
echo "inclass 'ibem'"                 # class of newly created image
echo "plev 1"                        # set perecentage level
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".IBEM.PS'" #PS file
echo "go cntr"                        # do contour plot of defined area
echo "go lwpla"                       # write plot to PS file

## plot slice through dirty beam at 0 degrees
echo "blc 426,513"                    # start point of slice
echo "trc 598,513"                    # end point of slice
echo "plev 0"                       # set percentage level
echo "outfile ' '"                    # don't write to a file, plot
echo "go slice"                       # calculate slice
echo "go sl2pl"                       # turn slice into a plot file
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".SL0.PS'" # PS file
echo "go lwpla"                       # write plot to PS file

## plot slice through dirty beam at 45 degrees
echo "blc 426,427"
echo "trc 598,599"
echo "plev 0"
echo "outfile ' '"
```

```
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1_6-23_3.sl45.ps'"
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".SL45.PS'"
echo "go lwpla"

## plot slice through dirty beam at 90 degrees
echo "blc 512,427"
echo "trc 512,599"
echo "plev 0"
echo "outfile ' '"
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1_6-23_3.sl90.ps'"
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".SL90.PS'"
echo "go lwpla"

## plot slice through dirty beam at 135 degrees
echo "blc 426,599"
echo "trc 598,427"
echo "plev 0"
echo "outfile ' '"
echo "go slice"
echo "go sl2pl"
#echo "outfile 'myarea:1a1_6-23_3.sl135.ps'"
echo "outfile 'myarea:"$ARRAY:r"_6"$DEC"_"$WAVELENGTH".SL135.PS'"
echo "go lwpla"
```

Appendix C– processConfigs.csh

```
#! /bin/csh

# File:    processConfigs.csh
# Author:  Steven Heddle, heddle@cyberdude.com
# Date:    1st December 1999
# Version: 1.0

# This script creates a batch file to be processed by BATER or aips, which
will then
# generate output (as described in the header for batCreate.csh) for array
# configurations and conditions as specified in the command line invocation.
# A range of configurations may be specified in a file containing a list of
# uvcon-formatted data files, to be analysed at a specified wavelength,
# and at 1 or more specified declinations.
#
# The script performs repeated calls to batCreate.csh, and adds the required
# preamble and postscript for batch processing.

# Invocation: processConfigs.csh arraylist output_batchfile wavelength
#                                           aips_disk dec1 dec2 ...
#
# arraylist is a text file containing the filenames of the uvcon-formatted
# datafiles for each configuration to be processed.

# Processing the output batchfile
#---------------------------------
# Require the following environmental variables to be set before running
BATER
# with the following usage BATER NEW < output_batchfile. More flexible
# use of data disks is possible by creating output_batchfile with aips_disk
set as 0
# (any available aips_disk) and replacing 'BATER' in the previous usage with
'aips'
#
# export "MYAREA"=/wine/alma/myarea
# export "ASWDATA"=/wine/alma/aswdata
# export "DA01"=/winf/aipsdata01
# export "DA02"=/wing/aipsdata02
# export "DA03"=/winh/aipsdata03
# export "DA04"=/wini/aipsdata04
#
# The following line may need to be executed to set some remaining variables
# (combined this and the above in .profile)
# pushd /home/aips; source .profile; popd

if (${#argv} < 5) then
    echo "Usage: processConfigs.csh <arraylist> <output batchfile>
              <wavelength (mm)> <aips disk> <declination 1> <declination
```

```
2> ..."
    exit
endif

set ARRAYLIST = $1
set WAVELENGTH = $3
set DISK = $4
@ DECNUM = ( ${#argv} - 4)
set ARRAYNUM =  `cat $ARRAYLIST | wc -l`
if(`cat $ARRAYLIST | wc -l` == 0) then
echo "No antenna arrays!...Surely some mistake?"
exit
endif

echo "#### "$ARRAYNUM" array configurations specified, along with"
echo "#### "$DECNUM" different declinations,"
echo "#### for which we will process both zenith snapshots and 6 hour
tracks......."
echo " "

echo "1" > $2
echo "AMANAGER" >> $2
echo "batque 1" >> $2
echo "batclear">> $2
echo "batch" >> $2
echo "version 'myaips'" >> $2
set ARRAYCOUNT=0

while ($ARRAYCOUNT < $ARRAYNUM)
set DECCOUNT=4

@ ARRAYCOUNT = $ARRAYCOUNT + 1

while($DECCOUNT < ${#argv})

@ DECCOUNT = $DECCOUNT + 1

./batCreate.csh `head -$ARRAYCOUNT $ARRAYLIST | tail -1`
$argv[$DECCOUNT]
               $WAVELENGTH $DISK >> $2
end
end

echo "endbatch" >> $2
echo "submit"   >> $2

echo "Finished writing batch file to " $2
```

Appendix D– Excerpt of example batch file created by processConfigs.csh

```
1
AMANAGER
batque 1
batclear
batch
version 'myaips'
get nuvco3mm
indisk 0
outdisk 0
aparm(1)=0
aparm(2)=0.3
aparm(3)=+0
aparm(4)=0
aparm(5)=0
aparm(7)=180
infile 'aswdata:100A1.UVC'
outname '100A1SN+0_3'
go nuvco
inname '100A1SN+0_3'
inclass 'nuvco'
inseq 0
bparm 6,7,2,0
dotv -1
go uvplt
bparm 0
bparm(6)=4
outfile 'myarea:100A1SN+0_3.UV.PS'
go lwpla
nboxes 128
axtype 'R'
outfile 'myarea:100A1SN+0_3.HIST.PS'
go uvhgm
go lwpla
nboxes 0
infile '100A1SN+0_3'
inclass 'nuvco'
inseq 0
docrt 0
xinc 1
ncount 2000
outprint 'myarea:100A1SN+0_3.TXT'
go prtuv
bchan 0
outname ' '
outdisk 0
in2name ' '
in2class ' '
in2seq 0
in2disk 0
selfreq 99931
go imagr
in2disk 0
inclass 'imap'
inseq 0
blc 426,427
trc 598,599
outfile 'myarea:100A1SN+0_3.IMAP.PS'
go cntr
go lwpla
inclass 'ibem'
plev 10
outfile 'myarea:100A1SN+0_3.IBEM.PS'
go cntr
go lwpla
blc 426,513
trc 598,513
plev 100
outfile ' '
go slice
go sl2pl
outfile 'myarea:100A1SN+0_3.SL0.PS'
go lwpla
blc 426,427
trc 598,599
plev 100
outfile ' '
go slice
go sl2pl
outfile 'myarea:100A1SN+0_3.SL45.PS'
go lwpla
blc 512,427
trc 512,599
plev 100
outfile ' '
go slice
go sl2pl
outfile 'myarea:100A1SN+0_3.SL90.PS'
go lwpla
blc 426,599
trc 598,427
plev 100
outfile ' '
go slice
go sl2pl
outfile 'myarea:100A1SN+0_3.SL135.PS'
```

```
go lwpla
get nuvco3mm
aparm(1)=0
aparm(2)=0.3
aparm(3)=+0
aparm(4)=-3
aparm(5)=3
aparm(7)=180
indisk 0
outdisk 0
infile 'aswdata:100A1.UVC'
.
.
.

outfile 'myarea:91A1_6+0_3.SL90.PS'
go lwpla
blc 426,599
trc 598,427
plev 100
outfile ' '
go slice
go sl2pl
outfile 'myarea:91A1_6+0_3.SL135.PS'
go lwpla
endbatch
submit
```