Manual Peeling in Obit

W. D. Cotton, December 12, 2017

***Abstract*—“Peeling” is the generic name for determining and applying a direction dependent gain to a limited portion of an image, usually to reduce the artifacts due to a particularly strong source. Obit imaging tasks have a quasi-automated “Peeling” facility which is triggered by the brightness of the offending source. This is not always adequate as the troublesome source is usually far out in the antenna pattern and causing artifacts in spite of a relatively low peak flux density. This memo describes a technique for allowing greater flexibility in dealing with a troublesome source. Example usage is given.**

***Index Terms*—imaging, interferometry**

I. Introduction

PEELING is a generic technique for applying direction dependent calibration to a portion of a field of view. This is usually to reduce the artifacts from a particularly strong source with direction dependent gain effects due to pointing errors, the rotation of an asymmetric antenna pattern or spatial variations in atmospheric phase. These effects produce apparent time varying changes in source brightness or position resulting in non convolutional errors in the image. These errors produce artifacts degrading the image. If the direction specific gain corrections can be determined and applied to the data, the level of artifacts can be reduced or eliminated. This memo evaluates such a technique using the Obit package [1] [1].

II. Peeling

The term “peeling” is used by a number of authors to mean slightly different things although usually some variant of determining and correcting the direction independent gains over some subset(s) of an image. In the following “peeling” will be used for the process of:

1) Image the full field of view.
2) Subtract the sky model excluding the region to be peeled.
3) Self calibration of the residual data to determine the corrupting gains wrt to the full field of view and a corrected image (sky model) of the source being peeled.
4) Subtract the model of the peel source from the data set applying correction for the corrupting gains.
5) Reimage the residual data for the non-peel portion of the sky model.
6) Restore the corrected peel sky model to the final CLEAN image.

A. Automated Peeling in Obit

Imaging tasks in Obit have optional peeling triggered by a peak in the image above some threshold (PeelFlux). The facet in which the peak is found is used to determine the peel target. The sky model from other facets are subtracted and a self calibration is performed using the parameters PeelFlux, PeelRefAnt, PeelSNRMin, PeelSolInt, PeelType, PeelMode, PeelNiter, PeelMinFlux, PeelAvgPol, and PeelAvgIF. The Peel model is then subtracted from the data and the residual data is CLEANed. If the peak in this CLEAN is still above the peel threshold, peeling is repeated until no residual peak above the threshold remains. The peeled sky models (CLEAN components) are then restored on the final image.

This process is often coupled with the “autoCenter” mode [2] which has two desirable properties here: 1) sources above a threshold (autoCen) will be centered in a new facet and 2) this facet is relatively small. This limits the region of the field of view being included in the peel. This scheme works when the source(s) to be peeled is (are) the brightest sources in the field of view which is frequently not the case.

B. Generic Peeling

It is useful to be able to specify the location and extent of the region to be peeled. A generic description of this process is

1) Initial image with direction independent self calibration. This will identify any source(s) needing peeling.
2) Copy self cal gain (AIPS SN) tables to the input dataset. If the input data is a multisource dataset (has AIPS SU table) then CLCal needs to be run to generate an AIPS CL table with the initial plus self cal gains. For single source files, copying the last SN table is adequate. NB if A&P self calibration is used in addition to phase only self cal, this is more complex.
3) Generate a sky model (CC table) excluding the peel region.
4) Subtract the sky model minus the peel source from the initial uv data applying the self cal gains.
5) Self calibrate the peel source only dataset producing a corrected image and gain table describing the differences between the direction independent gains and those for the peel source.
6) Subtract a corrupted version of the peel sky model from the data and replace it with an uncorrupted version of the sky model. This step is composed of
   a) Generate a direction independent self calibrated version of the full data set (“data”).
   b) Generate a model visibility data set (“model”) identical in structure to “data”.

National Radio Astronomy Observatory, 520 Edgemont Rd., Charlottesville, VA, 22903 USA email: bcotton@nrao.edu

[1] http://www.cv.nrao.edu/∼bcotton/Obit.html

   c) Invert (1/gain) the total peel self cal gain table copying to the "model" data and replace blanked solutions with (1,0). Replacing failed solutions with (1,0) helps deal with the baselines and frequencies for which the peel source is too faint to give valid solutions. When the peel source is too weak, using a gain of (1,0) causes little harm while flagging these otherwise usable visibilities could seriously degrade the final result.
   d) Applying the inverse corruption table to the "model" dataset, subtract it a visibility at a time from "data".
   e) Add the uncorrupted peel sky model back into "data".

7) Image the "data" dataset which should result in an image with greatly reduced artifacts.

C. "Manual" Peeling in Obit

The basic components of the process outlined in Section II-B, excluding the first 2 and the last, are implemented in python module python/PeelScripts.py.

1) SelectCC: The step in Section II-B bullet 3 is implemented in routine SelectCC. This generates a CC table excluding the region around position pos radius in extent. The documentation for routine SelectCC is given in Figure 1.

2) UVSub4Peel: The step in Section II-B bullet 4 is implemented in routine UVSub4Peel. This sets up for the subtraction of the the sky model missing the peel source which generates a new uv dataset. The routine returns a UVSub task interface object which may be further modified prior to execution. The documentation for routine UVSub4Peel is given in Figure 2.

3) ImagePeel: The step in Section II-B bullet 5 is implemented in routine ImagePeel. This sets up to the self calibration of the peel source only data set. A task object for MFImage is returned which may be further modified prior to execution. The documentation for routine ImagePeel is given in Figure 3.

4) SubPeel: The step in Section II-B bullet 6 is implemented in routine SubPeel. This routine subtracts the corrupted sky model of the peel source from a copy of the original data and then replaces it with an uncorrupted version of this data. Imaging this dataset will give the full field of view including the peel source but with reduced artifacts. The documentation for routine SubPeel is given in Figure 4.

III. Testing

A. EVLA data

This test used an extended synthesis of a field observed by the EVLA at 6 GHz. A bright region of emission is located near the edge of the beam and artifacts resulting from the varying brightness as the antenna beam rotates over the source are quite visible. The observations did not include pointing corrections and the resulting pointing errors may also contribute to the artifacts.

Imaging used Obit/MFImage with a field of view of 0.17° with outliers to 0.25° and a Briggs Robust factor of 0.0. Cleaning used 5,000 CLEAN components and 3 iterations of phase only self calibration were applied for the initial image. Peeling used 2 iterations of A&P with a 1 minute solution interval. The before and after peeling images of the field of view are shown in Figure 5.

B. MeerKAT data

The MeerKAT test is more difficult as it involves a very bright source near the first null in the antenna pattern and at part of the frequency range observed is at the null. This is very sensitive to pointing errors and both generates strong artifacts and renders the peel source quite weak over part of the bandpass. This leads to a poor model of the peeded source.

Imaging used Obit/MFImage with a field of view of 1.2° with outliers to 2°, a Briggs Robust factor of -1.0 and an inner UV taper of 400 λ [3]. Cleaning used 20,000 CLEAN components and 2 iterations of phase only self calibration were applied for the initial image. Peeling used 2 iterations of A&P with a 5 minute solution interval.

The region around the problem source before and after peeling is shown in Figure 6. The artifacts are greatly reduced but not completely eliminated.

IV. Discussion

A relatively generic technique for peeling a source showing artifacts is described and examples if its application are shown. Clear artifacts resulting from the varying antenna gain towards sources near the edge of the beam are greatly reduced.

Acknowledgment

I would like to thank the MeerKAT staff especially Sphesihle Makhathini and Fernando Camilo for assistance and for providing the MeerKAT data.

References

[1] W. D. Cotton, "Obit: A Development Environment for Astronomical Algorithms," *PASP*, vol. 120, pp. 439–448, 2008.

[2] W. D. Cotton and J. M. Uson, "Pixelization and Dynamic Range in Radio Interferometry," *A&A*, vol. 490, pp. 647–+, 2008.

[3] W. D. Cotton, "Inner UV Taper Weighting for Synthesis imaging," *Obit Development Memo Series*, vol. 50, pp. 1–9, 2017. [Online]. Available: ftp://ftp.cv.nrao.edu/NRAO-staff/bcotton/Obit/InvTaper.pdf

```
SelectCC(im, inCC, outCC, radius, peelPos, err)
    Select/copy CCs more than radius from peelPos

    This generates a CC table which can be subtracted from the
    uv data and remove all sources but the peel source area.
    * im     = Python Image with CC Tables
    * inCC   = input CC version
    * outCC  = output CC version
    * radius = radius (deg) of zone of exclusion
    * peelPos= [RA, Dec] in deg.
    * err    = Python Obit Error/message stack
```

Fig. 1. SelectCC function

```
UVSub4Peel(uv, source, im, inCC, err, nfield=1, doGPU=False, \
    gainUse=1, flagVer=-1, nThreads=1, noScrat=[0, 0, 0], taskLog=’’, \
    debug=False)
    Sets up for subtraction of non peel sources from data

    UV data should be have calibration tables from self calibration.
    Output data will be on the same disk as the input, seq=1,
    class=’4Peel’ and with the name of the source (up to 12 char)
    Returns UVSub task object
    * uv        Dataset to be subtracted from
    * source    source name
    * im        Python Image with CC Tables
    * inCC      input CC version, should have had the peel source CCs
                removed using SelectCC
    * err       Python Obit Error/message stack
    * doGPU     Use GPU if available?
    * nfield    Number of facet images
    * gainUse   CL (SN) table to apply, -1=> no cal
    * flagVer   FG table to apply, -1=> no flag
    * noThreads number of threads to use
    * noScrat   AIPS disks not to use for scratch
    * taskLog   Log file
    * debug     If True leave debug Input file in /tmp
```

Fig. 2. UVSub4Peel function

```
ImagePeel(uvsub, peelPos, err, nxy=512, Niter=1000, minFlux=0.001, \
    maxPSCLoop=2, minFluxPSC=0.01, solPInt=1.0, minSNR=3.5, \
    Robust=0.0, doGPU=False, nThreads=1, noScrat=[0, 0, 0], \
    taskLog=’’, debug=False)
    Sets up to image subtracted uv data from UVSub4Peel, self
    calibrate peel source.

    Only does A&P self cal
    Returns UVSub task object, output image "IPeel’, uv ’UVPeel’
    * uvsub     task object from  UVSub4Peel
    * peelPos   [RA, Dec] in deg of source to peel
    * err       Python Obit Error/message stack
    * nxy       Size in pixels of x,y
    * Niter     max number of iterations
    * minFlux   Min flux density first CLEAN
    * maxPSCLoop max number self cal loops
    * minFluxPSC min peak for self cal
    * solPInt    solution interval for self cal
    * minSNR     min SNR of self cal solutions
    * Robust     Briggs Robust factor
    * doGPU      Use GPU if available?
    * nThreads  number of threads to use
    * noScrat   AIPS disks not to use for scratch
    * taskLog   Log file
    * debug     If True leave debug Input file in /tmp
```

Fig. 3. ImagePeel function

```
SubPeel(uv, source, imp, uvp, err, flagVer=0, nThreads=1, doGPU=False, \
    noScrat=[0, 0, 0], taskLog=’’, debug=False)
    Subtract Peel model w/ solutions, then add back w/o corruptions

    UV data should have calibration tables from self calibration
    Output data will be on the same disk as the input, seq=1,
    class=’PelSub’ and with name = source (up to 12 char).
    Returns Peel source subtracted/replaced data
    * uv        Dataset with cal tables
                Needs at least the self cal gain table
    * source    source name
    * imp       Peel source model (CC table from ImagePeel)
    * uvp       UV data the result of peel (ImagePeel)
    * err       Python Obit Error/message stack
    * nThreads  number of threads to use
    * flagVer   FG table to apply, -1=> no flag
    * doGPU     Use GPU if available?
    * noScrat   AIPS disks not to use for scratch
    * taskLog   Log file
    * debug     If True leave debug Input file in /tmp
```

Fig. 4. SubPeel function

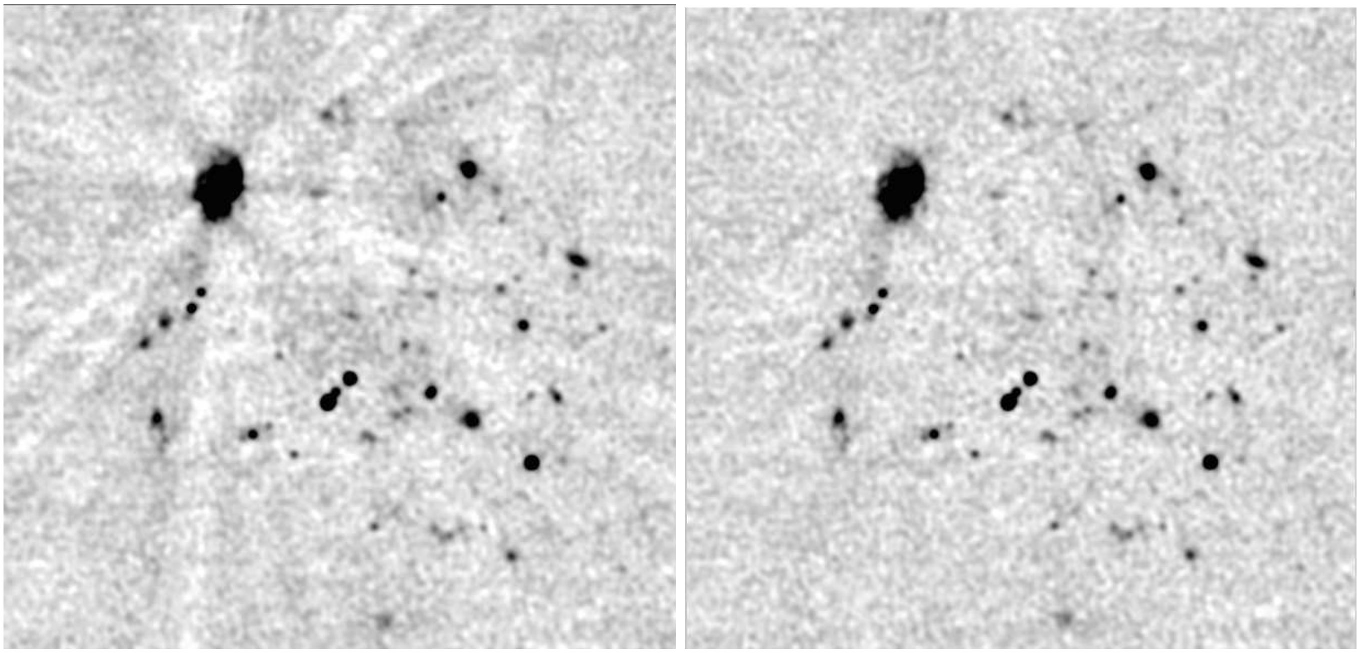

Fig. 5. Negative grayscale comparison of a field observed with the EVLA at 6.3 GHz before (left) and after (right) peeling. The stretch in both images is -20 to 100 μJy/beam and the field shown is 15’ × 15’.

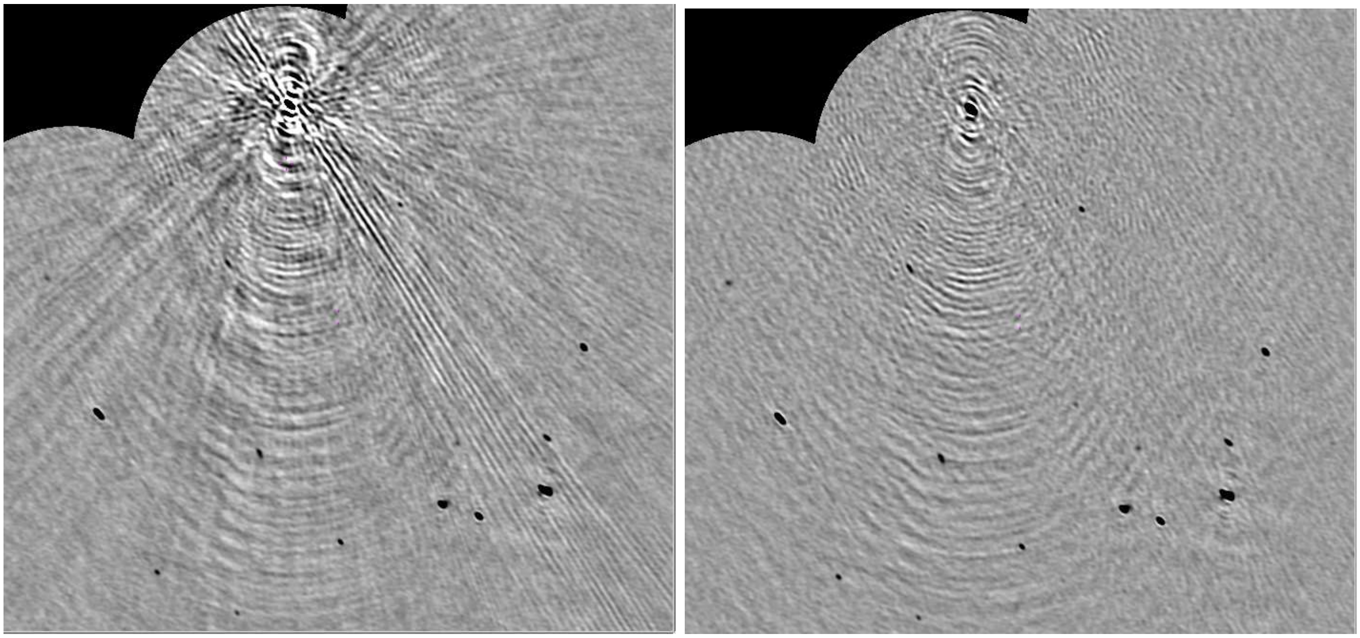

Fig. 6. Negative grayscale comparison of portion of a field observed with MeerKAT before (left) and after (right) peeling. The stretch in both images is -200 to 300 μJy/beam and the field shown is 13.8’ × 12.9’.