LBWG memo 22

Loop 3 in difmap

Neal Jackson, 2019.05.09

Summary. Loop 3 is the bottom-level program which receives a dataset, and performs imaging and selfcalibration. It generates an `h5parm` file containing the corrections, and passes it back to loop 2 for incorporation in the global phase model. Memo 16 describes the implementation using `wsclean` for the imaging and `NDPPP` for the calibration.

Difmap (Shepherd 1996) is a purpose built program for doing precisely this process for VLBI data. It is very compact and efficient, and is typically faster than most MS-based methods by about an order of magnitude. It has a number of disadvantages: it is relatively rigid, and concentrates on doing a few things well; it writes out images and corrected data but not corrections; and it does amplitudes and phases, but not delays.

Implementation of loop3-difmap

There is a test version of loop3 difmap code available on `github.com/nealjackson/loop3_difmap` . It consists of a main program, `loop3_difmap.py` and two service routines, `corplt.c` and `find_difmap_chan.py` . The latter two are there to solve features of `difmap` which make it difficult to incorporate in scripts.

The most fundamental problem is that `difmap` will not write out phase and amplitude solutions, but only plot them with the interactive command `corplot` (or non-interactively, will write a Postscript file containing the plots). In order to run `loop3_difmap`, you will need to reinstall the `difmap` distribution:

- Perform the `configure` step as usual.
- Replace the file `corplt.c` in the `difmap_src` directory, off the main directory, with the revised version from the github. The difference in the latter version is that amplitude and phase corrections are written from each station on the `corplot` command, in addition to being plotted. Writing is done to a file called `CORPLT` in the current directory, which is then read by the Python script after the CLEANing/self-calibration is finished.
- Perform the `makeall` in the distribution directory as before.

Once this has been done, you can run the test version using the command

```
python loop3-difmap.py [FITS filename]
```

The program performs the following steps:

- For the XX polarization, write a difmap script to process the FITS file. This involves specifying the polarization and which channels to read using the `select` command. This is important because `difmap` will crash if it is given any channel which is all flagged, so there must be a file with name `dchan_[FITS filename minus .fits` which contains the command `select I,C1,C2,C3,C4....` where good channels are from `C1-C2`, `C3-C4` etc. If this file exists, the select command will be read from it and the I polarization substituted for XX. If not, it will be generated by reading the file into AIPS and using a `parseltongue` script to find the empty channels. For a given field in which many sources have been split, you should be able to use the same channel numbers for each source.

- Fill in the rest of the script using some fairly obvious imaging parameters (image size, pixel size etc) in the arguments. Note that *the script uses the startmod command*, so it will initially self-calibrate to a point source. This will create non-existent sources in regions of blank sky, so should be replaced with `rmod` if a model is available (in which case a script needs writing to translate model files from sourcedb to Difmap model format) or by doing an iteration of CLEAN first.

- Three rounds of phase selfcal are done with different u-v weighting schemes. Clean-selfcal loops proceed until the peak residual drops below a set fraction of the rms noise. In this section and the next the algorithm is based on `VLBI_imaging.py` by Kistof Rozgonyi (`github.com/rstofi`).

- An overall amplitude scaling is then performed followed by amplitude selfcalibration with steadily decreasing solution intervals, and intervening phase selfcal if the signal-to-noise of the output drops.

- A lot of outputs are generated, including the corrected dataset, the CLEAN map and a plot of the corrections. The modified `corplt.c` also generates an ASCII table of phase and amplitude corrections, which is copied to a file `CORPLT_XX`.

- The process is repeated for the YY polarization to give another set of outputs including `CORPLT_YY`.

- Both correction plot files are read in and converted to an `h5parm`, together with antenna arrays derived from the `corplot` output. There are significant shortcuts:

  - It is assumed that the number of solutions in time is the same for both polarizations (if not true there will be a crash)
  - The UTs are written in the format written out by difmap - seconds since the start of the day - rather than as MJDs.
  - Solution weights are all 1. There should be some editing of the `h5` file to adjust the weights and/or to edit bad points.
  - I haven't checked if the solutions in the h5 file apply to the data in the original measurement set. (I don't see why they shouldn't, but there may be problems with phase sign conventions etc which need to be tested).

Operation

I have tested the algorithm on one dataset so far. It runs of the order of 20 clean/-selfcal loops in approximately 1 minute, compared to 30 minutes for 6 loops of the `wsclean/NDPPP` route of the standard loop 3 (though NB the images are only 512 on a side rather than 4096 - this benchmarking should be done more carefully). It also produces better images (Fig. 1).

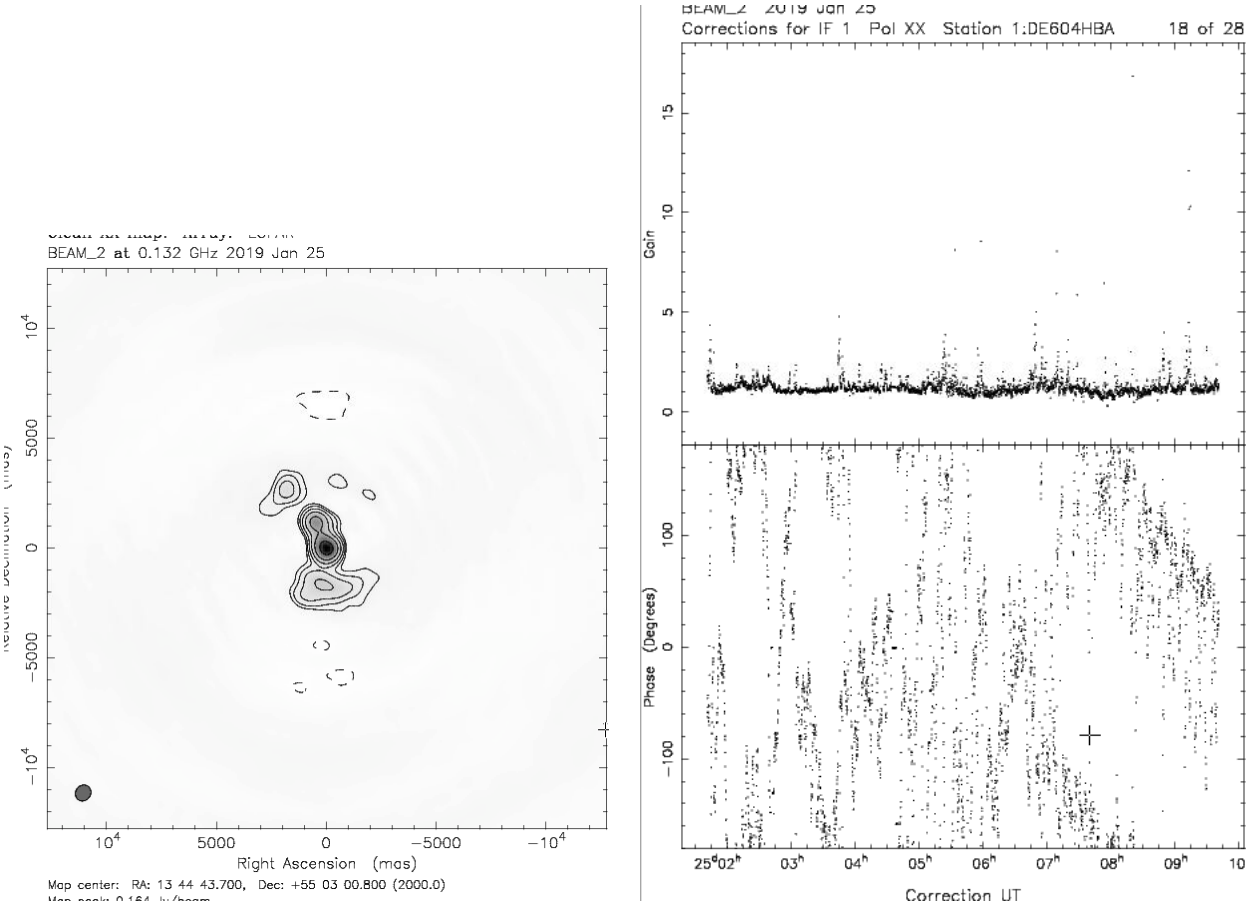

Figure 1: Maps of 1344+5503 (see memo 16) and difmap-derived corrections