A Simple Guide to Getting Started with DIFMAP

Dan Homan

1. Introduction

This is an outline of a bare-bones procedure for imaging and calibration in DIFMAP. It is far from complete and simply designed for just getting started in difmap. As an alternative to this document, the Difmap Cookbook by Greg Taylor is an excellent place to start. Before starting your first calibration, I strongly recommend reading Craig Walker’s chapter from “Very Long Baseline Interferometry and the VLBA” on “Practical VLBI Imaging”.... much of my approach to calibration and imaging comes directly from the advice he gives there.

There are three main things you will be doing here

- **Editing ('Flagging') Data**: This simply involves throwing away clearly bad data. The bad data are usually associated with specific times on specific antennas, so 'antenna-based' editing is usually the best approach. Most data edits are permanent, but occasionally one will edit (or ‘flag’) a particular antenna/time range initially with the idea that it may be recoverable later through calibration once a good model is built.

- **Building a Model**: The interferometer data is a sampling of the Fourier transform of the sky brightness distribution. Because the (u,v)-plane is discretely sampled, a direct Fourier inversion will produce a large number of artifacts in the image plane. We therefore need to create model of the sky brightness distribution by either (1) cleaning the source using the well-know 'CLEAN' algorithm to de-convolve the image, (2) fitting model components directly in the (u,v)-plane, or (3) some combination of cleaning and uv-modeling. The clean algorithm is by far the dominant approach, but there are situations where constraining some of the flux through uv-modeling is extremely useful (example: closely spaced core components which CLEAN cannot resolve). The end result of this process is a list of model components from which calibration can be performed or images can be made.

- **Calibrating Data**: Calibrating the data is accomplished by comparing the data to the model and looking for antenna based corrections which reduce the difference between the two. As the model is usually derived from the data prior to calibration, this approach is called 'self-calibration'.

In practice, the self-calibration process involves may cycles of model-building and calibration. Usually one will CLEAN as much believable flux as possible to create a model then calibrate against that model. The old model is then thrown away and another is built from scratch with the
newly calibrated data. This new model can then be used to further refine the calibration and the process repeats until no further improvement can be found.

Here I describe a simple approach to this process using the Difmap software package. What follows assumes you have a data set to get started on. In *italics* are asides which can be skipped on a first reading.

2. Starting DIFMAP

You will want to place the data you plan to work on in its own directory. You can start DIFMAP by typing “difmap” in that directory. The program will start up and you will be given a prompt at which you can enter commands.

0> observe filename

This command will load your data and attempt to determine the integration time from the data itself. You can specify an integration time by typing ‘observe filename, 2’ where “2” is a 2 second integration time. For more information on any command, use the “help” command, e.g. ‘help observe’

Next you will want to ‘select’ the data you wish to work on

0> select i

This command selects the Stokes I data. (Other stokes parameters [Q,U,V] as well as the pure correlations [RR, LL, LR, RL] can be selected instead.)

0> uvweight 0,2

This command sets the weighting which the data will receive when we do the Fourier transform of the uv-data into an image. The weighting chosen above is “natural” weighting and weights the data only based on their SNR. (If the SNR of all the data is the same, then all the data points would be treated equally). There are other weighting choices we could use, if the data require it. For example ‘uvweight 2,0’ would be a “uniform” weighting where each bin in the (u,v)-plane of binwidth 2 is given the same weight. This will effectively down-weight short-baselines (where there are more data per bin) relative to the longer baselines. The effect of uniform weighting is to give higher resolution.

Aside: Difmap allows you to change your data weighting schemes on the fly, as you are cleaning a source. This is extremely useful. If you have a source with some compact structure, you can start a cleaning cycle by cleaning with uniform weighting. As the residuals in the map get weaker, you can then switch to lower resolution by using a smaller bin size or going to natural weighting. Finally, if there is still flux to be cleaned, but the residuals are very low, you can add a ’uv taper’ which will down-weight the long baselines.
As a normal course of action, I usually start with some combination of uniform and SNR based weighting... something like ‘uwweight 2,1’. As I clean more flux and the residuals become weaker, I switch to natural weighting ‘uwweight 0,2’. Finally if there is still flux to be cleaned, but the residuals are too weak in natural weighting, I will try a ‘wetaper’ to get the rest of the flux.

Once the data are pretty well calibrated, I sometimes find I am limited by the fact that clean seems to do a poor job on some very compact core structure. In such a case, I might try ‘super-uniform’ weighting (using a large bin size like 10 or 20) for the first couple of clean iterations, or I will simply ‘modelfit’ the core with a small number of delta-functions instead of using ‘super-uniform’ weighting.

0> mapsize: 1024,0.1

This command sets the map size to 1024x1024 pixels with cellsize of 0.1 milli-arcseconds per pixel, appropriate for 15 GHz MOJAVE data.

3. Examining and Editing the Data

0> radplot

This command will allow you to see all the data at once versus uv-distance. PGPlot will ask you for a device, and you should specify ‘/xs’ or ‘/xw’ to get an X-window with the data displayed inside. You can now work in this window, examining and doing a rough edit of the data. Type ‘h’ to get a list of possible commands. You can edit data points by simply clicking on them. I recommend not editing here, but using this plot as a diagnostic to see where the bad data lie. The ‘n’ key will allow you to highlight each antenna in turn. This is very useful for identifying which antennas are associated with bad data. The ‘v’ key will display the specific baseline, time information for a given point, and is very helpful for finding what time the bad data lies in. Finally, the ‘x’ key will exit the plot window and return control to the dmap console.

Aside: Occasionally one antenna will be systematically off from the others at the same (u,v) distance... usually with a significantly lower amplitude. If such an antenna exists, you will spot it here in this plot. It is a wise idea to prevent such an antenna from contributing to the initial model building. You can deal with such a situation by flagging the offending antenna (use vplot or tplot), build a reasonable model in the usual way (described below), then bring the antenna back and calibrate it by itself (using the ‘selfant’ command to fix all the other antennas).

0> projplot 0

This command is very much like “radplot”, but the data are plotted versus projected uv-distance in a particular direction (0 degrees). You can change this direction by using the “<” and “>” keys. This task is very useful for examining the uv-data along the structure axis of the source. Sometimes features of the data can be obscured by radplot but are clear here.
0> vplot 3

This command is meant for a more careful examination/editing of the data. If only the phases are plotted, you will want to see both amplitude and phases so press '3' followed by 'n' to replot with both amplitudes and phases. As before 'h' will list the possible commands. Useful ones are 'n' and 'p' which show the next/previous set of baselines; '[' and ']' step through the IFs. 'i' toggles editing on one IF or all IFs. 'space' toggles between antenna and baseline based editing. You can edit a region with the 'c' command which allows you to define a box for editing. You can restore a region with the 'r' command. Individual points may be edited by simply clicking on them.

4. Self-calibration and Imaging

0> mapl

This command generates an image by Fourier transforming the uv-data. Most VLBI sources will show a bright central point and a dirty beam pattern around it. Again, there are a large number of commands you can use in the window. Some of the most common are: simply clicking with the mouse defines the corners of a clean box where clean components will be allowed, repeatedly pressing “b” puts you in fiddle mode where you can change the grey scale by moving the mouse around, “z” turns on zoom mode - now clicking the mouse will define a window to zoom into, “m” toggles display of the model/clean components.

4.1. Clean Boxes

You will currently have no clean boxes defined, so you need to define one. Repeatedly press the “b” key and move your mouse to change the grey scale so that only the brightest part of the image is showing - basically we don’t want to see any beam effects. Now use your mouse to put a (reasonably tight) clean box around the brightest part of the image.

Clean boxes define the regions where clean components can be put by the program when we ask it to clean the image. The collection of clean components that we find will become our model and be used to self-calibrate the data. Because we will use the clean components for calibration, we don’t want clean components to go just anywhere in the image. We only want clean components in regions we trust as being real parts of the source.

As we clean the image, the brightest parts will be cleaned away (including their nasty beam effects) leaving the less bright parts of the source. As these parts become more prominent, we will put clean boxes around them as well. Eventually, we will have a complete model of the source.
4.2. Cleaning

$0>\text{clean 50, 0.05}$

This command will clean 50 components with a loop gain of 0.05. The clean components are chosen at whatever is the brightest point in the image. The loop gain determines how strong the subtracted clean component will be. At a loop gain of 0.05, the subtracted clean component will be 1/20th the strength of the brightest point in the image for that iteration. This will assure that the components will be subtracted gently from the image. As the calibration of the data improve, larger numbers of clean components can be taken in a single clean command. After each clean command, it is a good idea to examine the result...

$0>\text{mapl}$

Now the image will show the residual flux... i.e. the clean components from the previous command have been subtracted from the data. You can see where the clean components were taken by pressing "m". The clean components are marked by little cross-hairs (any components taken from the same pixel in the image are merged into a single component). Red cross-hairs indicated negative clean components. You will generally want to delete any clean components that are negative or that appear to close to the edge of your boxes. Use the "r" key to remove the clean component nearest the cursor from the model.

4.3. Self-Calibration

With this initial clean, we have a partial model of the source. DIFMAP allows us to use these partial models to do phase calibration against the remaining data. (In AIPS we would build a full model of the source before doing any self-calibration.) Amplitude self-calibration must wait until we have a full model of the source.

To do a phase selfcal, simply use the command:

$0>\text{selfcal}$

4.4. Examine the Data

Now use radplot, projplot, or vplot to examine the data. You will see that the Fourier transform of the model (as far as we have one) is displayed along with the data for comparison. As we clean more flux, self-calibrate the data and improve the model, the agreement should improve considerably. These tasks should be repeatedly run to assess the agreement between model and data (in both amplitude and phase) and look for problems in the process. See the notes below about how to 'un-calibrate' the data and the usefulness of comparing the uncalibrated data to the best model at any given time.
4.5. Repeat!

Go back to the mapplot step and repeat. Make new clean boxes if necessary. Change the weighting with the 'uvweight' command if necessary. Clean some more flux. Examine the clean components with mapplot. Self-calibrate (in phase). Examine the data. Repeat.

4.6. Amplitude Self-Calibration

When you have cleaned as much flux as possible and made the model as complete as possible, it is time for an amplitude self-calibration. The command to self-calibrate in amplitude is ...

0> selfcal true,true,1e6

This command will self-calibrate the data in amplitude on a 1 million minute time scale. This time scale is simply chosen to be much longer than the experiment, so that only a single correction is applied at each antenna. The command 'gcale' accomplishes the same thing, but it puts in a normalization factor which assumes that the average amplitude gain in the experiment is 1.0. Sometimes this assumption is desirable and other times it is not. It is designed to prevent overall loss of amplitude in the experiment due to not having enough flux in the model prior to calibration. I prefer to use the 'selfcal' command and to ensure that I have cleaned all the possible flux before calibrating in amplitude. If it is simply not possible to clean all the flux (as judged by the agreement between the data and model on the short spacings), the command 'selftaper' can be used to prevent the short spacing from contributing to the calibration solution.

Aside In future iterations, the interval of amplitude self-calibration can be shortened as confidence in the model increases. I usually go next to one correction every 1 or 2 hours, eventually getting to one correction per scan. When the model is extremely good, and the data SNR are high, it is reasonable to do a final amplitude correction on a point by point basis to correct the very short term fluctuations.

To examine the corrections that have been applied to the data set (since you loaded it), use the command “corplot”.

0> corplot

Also, you can use the command:

0> mapplot cli

to have a look at your image with the clean components added back in (without their annoying beam affects). The 'loglevs' command can change the contour levels. A useful thing to do in this plot is to make a reasonable sized clean box well away from the source. Then with your cursor in this box, press the 's' key. You will get statistics for the image noise in this box. Delete this extra clean box using the 'd' command so that real clean components won’t go there in the future.
4.7. Starting Fresh using the Calibrated Data

The above process is usually repeated several times. Each time using the improved calibrated
data from the previous run.

0> wmod templ.mod (saves a copy of the current model)
0> clrmod true,true,true

This last command will clear your current model (Your clean boxes will stay in place, so
you won’t need to modify them, unless something new pops up.) Now repeat this whole process
of cleaning and self-calibrating. This time it is probably ok to finish with a shorter amplitude
self-calibration time scale.

In general, you will continue these process of clearing out the old model, cleaning to get a new
model, and calibrating against the new model, stopping when no further progress can be made.

Aside Once the new model has been built, it is often useful to clear out the old calibration....

0> uncal true,true

This command will remove all the calibration that has been applied up to this point (since the
data were loaded). Because we still have the model, we can just re-calibrate – first in phase then
in amplitude. Getting “fresh” calibration solutions like this can prevent sinking into local minima
in the space of possible calibration solutions.

It can also be extremely useful to compare (using radplot, projplot) the uncalibrated data to
the best model at that time. From such a comparison it is often clear if a serious divergence from
the original data has occurred. For example, you might notice that the total flux in the 'best model'
is well below the flux in the uncalibrated data – indicating that an amplitude self-cal was probably
done before all the flux was cleaned.

5. Saving the Results

Once you think you’ve made as much progress as you can, it is useful to write the data and
model out to disk....

0> wmod filename.mod

To write out the best model. This is a text list of clean component and any uv-modelfit
components in a single file.

0> wmap filename.map

To write a FITS image of the best model. This is not a 'final' image, as it has not been deeply
cleaned, but it is a FITS image of the best self-calibration model. It is useful to have this if you
wish to use this model in AIPS for calibration (such as CALIB or LPCAL). The model list is just a CC table attached to this file.

Next, I un-calibrate the data and write that out. I like to have edited but \textit{uncalibrated} data, because I can pick up this process again without worry about what calibration might be applied to the data.

0> uncal true,true
0> wobs filename.edit

It is also useful to save the windows “wwin filename.win”. All of these thing can be read back into difmap to pick up the process where you left off... e.g.

0> observe filename.edit
0> select i
0> rmod filename.mod
0> rwin filename.win
0> selfcal
0> selfcal true,true, time (for whatever timescale)

6. Making a ‘Final’ Image

With the data fully calibrated, one should be able to CLEAN from scratch with very little restriction on the location of clean components.

0> chmod true,true,true

To clear your calibration model.

0> mapl

You will see the familiar Fourier transform of the data with all the nasty beam effects included. Now use your mouse to make a very large clean box that covers 75\% of the image. This box will allow clean components to be put essentially anywhere.

0> clean 500, 0.05

If at the end of this clean cycle there is still positive flux being cleaned, clean even more components until positive flux is no longer being accumulated. It is ok (and perhaps even a good thing) to push this final clean well into the noise.

0> mapl cln
To examine your final image. The contour levels may not be low enough. You can adjust them with the ‘loglevs’ command.

0> loglevs 0.3, 100, sqrt(2)
then
0> mapl chn

If you would like to save your image as a post-script file. Do another “mapl chn” and zoom to the frame that you like best. Press “k” to save this framing. Now exit mapplot and type

0> mapcolor none
0> device file.ps/ps
0> mapl chn

Now your image will be saved as “file.ps”, and you can print it later. To change these settings back, type

0> mapcolor grey
0> device /xs

7. MOJAVE Specific Short-Cuts and Procedures

Matt has defined a series of macros for Difmap which make the process of calibration and editing in Difmap much faster and easier for MOJAVE data. These macros also simplify the process of saving the data to be sure then end up in the format we want. What follows is a very brief description of each of these commands in the approximate order you would use them. I’ll also describe the steps above they would replace.

0> observe source data file
Just reading in the data as above.
0> medt

This is a macro for editing the data. It will pop-up a series of radplots which will show you sequentially the 'V','Q','U', and then 'I' data. You can edit a few high (or low) points here by clicking on them, but I really recommend using these plots to find where the bad data is (using 's' and 'n' to find specific times and antennas) and then doing antenna based flagging in vplot later. Usually it is best to flag whole scans of a given antenna (perhaps all IFs or just one, depending on the cause of the problem). This is because there may be other bad data here which does not show up clearly in the radplot, but would still produce poor results in the polarization images.
The 'medt' command will also set the mapsize and uvweighting for you. Note that it sets the uvweighting to '2,-1' which is uniform weighting. As you clean, you will eventually switch to uvweight 0,-2 which is natural weighting to clean the more extended structure.

0> vplot 3

To do the editing described above. The most common source of problems is low elevation on the boundary antennas (SC, HN, MK) on either the first or last scan. These scans will often appear either noisy in V,Q,U, and/or at lower amplitude in I from the radplots. I've found it is best to simply flag these scans entirely on these antennas rather than try to resurrect data which is doomed to simply degrade the polarization images anyway.

An antenna which is systematically low does not always have to be flagged forever (although if the factor is a larger than 50 percent, it will cause problems for polarization, especially stokes-V, down the line) and can be brought into line in the fashion I described in the earlier parts of this document.

0> mav

This command averages the data to 10 second integrations and will replot the 'V','Q','U' and 'I' data so you can get an idea if there is any bad data remaining that needs to be flagged. If so, I recommend the 'vplot' command again, as described above. Special Note: the 'mav' command should only be run once (you cannot run it again without causing major problems).

0> mapl

Now plot the data and begin the process of assigning clean boxes, cleaning, self-calibrating, and cleaning some more as described in the rest of the document above. So a reasonable set of commands might be...

0> clean 50, 0.05

0> mapl (to check that the components cleaned are OK, and remove any that aren't)

0> selfcal (just selfcal in phase only)

0> mapl (to see the effect of the selfcal, and maybe add a clean box if nec.)

0> clean (will clean again with the last setting, 50 components and a gain of 0.05)

0> mapl (to check the components again)

0> selfcal

etc... I do this until some extended flux begins to show, then I switch to natural weighting

0> uvweight 0,-2

Now repeat the above mapl, clean, mapl, selfcal cycle until you've cleaned all that you can.
In some cases, I introduce a 'uv taper' to get even more extended flux, but this is not necessary most of the time, and is a judgement call.

Once we have all the flux (check with 'radplot' or 'projplot'), an amplitude selfcal is OK...

0> selfcal true,true,1e6

See the parts above for how to handle cases where you don’t have all the flux by using the 'gscale' command (or better yet) the 'selftaper' command.

0> wmod temp.mod (save a current version of model, for safety)

0> clt

This command is another macro, and it deletes the current model, resets all the tapers, and puts the weighting back to uniform (uvweight 2,-1). Now all is ready to repeat the above cleaning process with the improved data, but this time though you can be much quicker (now that the boxes are set and you know about how many clean components are necessary before you need to switch weighting schemes)... For example, I might use

0> clean 200; uvweight 0,-2; map

Note that I’ve strung three commands on one line, which can be a useful timesaver. The above assumes that 200 cc are about right for uniform weighting, if that is true, it is OK to just clean that many before switching to natural weighting. No selfcal should be necessary yet, so if the map looks OK, just keep cleaning until it looks like it needs a selfcal. Then selfcal in phase (just 'selfcal') and keep cleaning until you’ve got all the flux. Note that you should do at least one selfcal in phase alone before selfcaling in amplitude again, so

clean clean clean (maybe a selfcal or two) then

0> selfcal (to be sure you gotten a phase selfcal in)

0> uncal false,true (to remove the previous amplitude selfcal- usually a good idea)

0> selfcal true,true, 60

Now save the model ('wmod temp.mod'), clear it out ('clt'), and repeat the above sequence to get to an even shorter amplitude selfcal. I usual cycle the amplitude selfcals down from 1e6 (1 million minutes) to 60,5,1, and finally unconstrained (just selfcal true,true). Note that in the later stages, you definitely want to clear out the previous selfcal solutions (using the above uncal command) before doing the very short self-calibrations in amplitudes.

The reduced 'chisq' after each amplitude selfcalibration should be dropping appreciably until it is somewhat smaller than unity for your final calibration. Once you have are happy with the calibration it is time to save it, and all of the files we need for the rest of the MOJAVE calibration. Use the 'msave' command for this...
0> msave source.name

This command will produce several files...

source.name.uvfraw - edited, uncalibrated data (important that it has no calibration applied)
source.name.CMAP - FITS image of the calibration model
source.name.CMOD - difmap version of the calibration model components

You are now technically done, and could exit Difmap, but I think it is a good idea to check the quality of the calibration by making a very deep image. To do this you will need to recalibrate the data (the above command un-did all the calibration)

0> selfcal true,true

Next you will do a 'final_clean'

0> @../final_clean i

This will call and run the final_clean script for stokes I. After it finishes, examine the image with the 'map l ch' command to see if it looks comparable in quality to other MOJAVE epochs. The SNR should be many thousand to one.

If all is OK. Quit difmap and rename the log file 'mv difmap.log source.name.log'