
GMRT tutorials

Release 0.1

Ruta Kale

Mar 22, 2023

CONTENTS:

1	Prerequisites	3
1.1	Interferometric data	3
1.2	Pulsar/Beam data	3
2	Introduction to the GMRT data	5
2.1	Continuum and spectral line data	5
2.2	Pulsar/Beam data	5
3	Continuum data reduction using CASA	7
3.1	Introduction	7
3.2	Data inspection	8
3.3	Flagging	9
3.4	Absolute flux density calibration	13
3.5	Delay and bandpass calibration	13
3.6	Gain calibration	14
3.7	Transfer of gain calibration to the target	15
3.8	Splitting the calibrated target source data	17
3.9	Flagging on calibrated data	18
3.10	Averaging in frequency	18
3.11	Imaging	19
3.12	Self-calibration	21
4	Spectral line data reduction using CASA	25
5	Pulsar data analysis	27
5.1	Introduction	27
5.2	Searching for new pulsars	29
5.3	Timing of Pulsars	37
5.4	Acknowledgements and Credits	46
6	CAPTURE Pipeline	47

You can learn GMRT data analysis using the tutorials provided here.

PREREQUISITES

1.1 Interferometric data

The interferometric (continuum/spectral line) data from GMRT can be analysed using NRAO's Common Astronomy Software Applications ([CASA](#)) or Astronomical Image Processing System ([AIPS](#)). We will focus on the analysis using CASA in these tutorials.

1.1.1 CASA

You need to download and install CASA. It is important to read the [CASA documentation](#) in order to understand the details of the CASA tasks and the overall functioning of CASA. It will be useful to get familiar with the [usage](#) at the ipython prompt before starting the tutorials. The specific CASA version in which the tutorial was prepared is provided at the beginning of the individual tutorials.

1.2 Pulsar/Beam data

The pulsar/beam data are typically recorded using the phased-array (PA) and the incoherent-array (IA) observing modes of the GMRT. The format of the native beam data is same for both the PA and the IA observing modes. After a simple format conversion using [SIGPROC/RFIClean](#), the native beam data from GMRT can be used with the publicly available softwares such as [SIGPROC](#), [PRESTO](#), [DSPSR](#), etc. The tutorial will use data already converted to the SIGPROC filterbank format. The primary softwares needed for the tutorial are [PRESTO](#), [tempo2](#), and their dependencies.

INTRODUCTION TO THE GMRT DATA

GMRT can record data in continuum, spectral line and pulsar/beamformed (phased/incoherent array, primarily to observe pulsars/transients) modes.

2.1 Continuum and spectral line data

These are data typically recorded with the full array and contain visibilities. The raw data are in the “lta” format. The continuum and spectral line data will both be in the same format. The data in lta format is converted to FITS format using the in-house codes *listscan* and *gyfits*. The FITS file is further converted into the measurement set (ms) format in order to analyse the data using CASA.

The GMRT online archive (GOA) allows you to download data in FITS and lta formats. Thus if you downloaded your data in FITS format you can skip the step of converting the format from lta to FITS.

The files downloaded from GOA having ‘gsb’ in their filenames are the data recorded with the narrow band backend which was called the GMRT Software Backend (GSB). The files with ‘GWB’ in their names are the data recorded using the GMRT Wideband Backend (GWB) which is the new wideband system (Upgraded GMRT).

2.2 Pulsar/Beam data

These are data typically recorded after adding the signals from multiple dishes or the full array, either incoherently (i.e., the incoherent array mode, IA) or coherently (i.e., the phased-array mode, PA). The native data are in filterbank format, i.e., a spectrum with the specified number of frequency channel is recorded for every time sample, for a specified sampling time. The data recorded by GSB and GWB are of same format.

The native beam (PA/IA) data from GMRT need to be converted to SIGPROC filterbank format, after which it can be used with softwares such as **SIGPROC**, **PRESTO**, **DSPSR**, etc. The format conversion can be performed using the `filterbank` command from **SIGPROC**. Alternatively, one can use the `rficlean` command from **RFIClean** to do the format conversion, in addition to mitigating some radio frequency interference (RFI). If desired, `rficlean` can also be used only for the format conversion, i.e., without performing any RFI excision.

CONTINUUM DATA REDUCTION USING CASA

The tutorial is intended to provide you an introduction to the basic steps involved in the analysis of continuum data from the GMRT. We will be using a band-4 (550 - 750 MHz) dataset.

CASA version used for the tutorial: 6.5.2

3.1 Introduction

You need to have CASA installed on your machine. You also need the data to be available on your disk.

From the GMRT online archive you can download data in “lta” or “FITS” format. If you downloaded the data in lta format then you will need to do the following steps to convert it to FITS format. You can download the pre-compiled binary files “listscan” and “gvfits” from the observatory.

3.1.1 LTA to FITS conversion

```
listscan fileinltaformat.lta
```

At the end of this a file with extension .log is created. The next step is to run gvfits on this file.

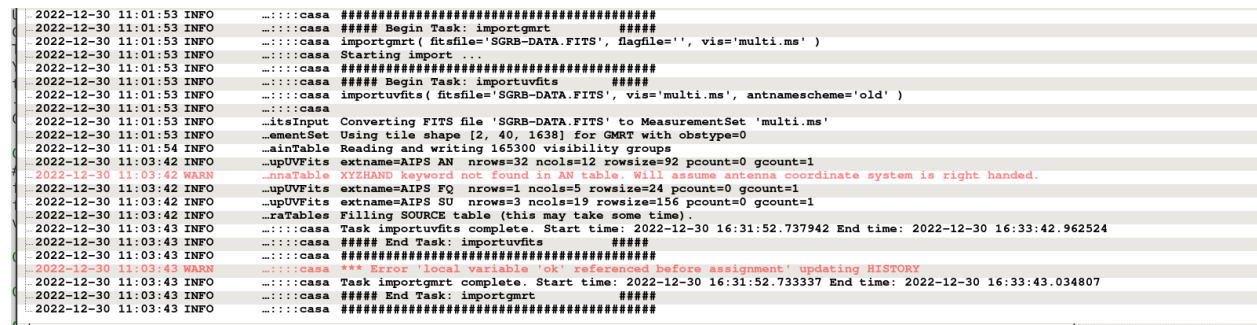
```
gvfits fileinltaformat.log
```

The file TEST.FITS contains your visibilities in FITS format. Before running gvfits, you can edit the fileinltaformat.log and provide a name of your choice in place of TEST.FITS.

3.1.2 FITS to MS conversion

At this stage start CASA using the command casa on your terminal. You will be on the Ipython prompt and a logger window will appear. The remaining analysis will be done at the CASA Ipython prompt. We use the CASA task importgmrt to convert data from FITS to MS format. In our case the FITS file is already provided so we proceed with that FITS file.

```
casa
inp importgmrt
fitsfile='SGRB-DATA.FITS'
vis='multi.ms'
go importgmrt
```



```

2022-12-30 11:01:53 INFO .....casa #####
2022-12-30 11:01:53 INFO .....casa ##### Begin Task: importgmt #####
2022-12-30 11:01:53 INFO .....casa importgmt( fitsfile='SGRB-DATA.FITS', flagfile='', vis='multi.ms' )
2022-12-30 11:01:53 INFO .....casa Starting import .....
2022-12-30 11:01:53 INFO .....casa ##### Begin Task: importuvfits #####
2022-12-30 11:01:53 INFO .....casa importuvfits( fitsfile='SGRB-DATA.FITS', vis='multi.ms', antnamescheme='old' )
2022-12-30 11:01:53 INFO .....casa
2022-12-30 11:01:53 INFO .....casa Converting FITS file 'SGRB-DATA.FITS' to MeasurementSet 'multi.ms'
2022-12-30 11:01:53 INFO .....casa MeasurementSet
2022-12-30 11:01:53 INFO .....casa Using tile shape [2, 40, 1638] for GMRT with obstype=0
2022-12-30 11:01:54 INFO .....casa Reading and writing 165300 visibility groups
2022-12-30 11:03:42 INFO .....casa extname=AIPS AN nrows=32 ncols=12 rowsize=92 pcount=0 gcount=1
2022-12-30 11:03:42 WARN .....casa .....nnaTable XYZHAND keyword not found in AN table. Will assume antenna coordinate system is right handed.
2022-12-30 11:03:42 INFO .....casa .....upUVFits extname=AIPS SU nrows=3 ncols=19 rowsize=156 pcount=0 gcount=1
2022-12-30 11:03:42 INFO .....casa .....Tables Filling SOURCE table (this may take some time)
2022-12-30 11:03:43 INFO .....casa Task importuvfits complete. Start time: 2022-12-30 16:31:52.737942 End time: 2022-12-30 16:33:42.962524
2022-12-30 11:03:43 INFO .....casa ##### End Task: importuvfits #####
2022-12-30 11:03:43 INFO .....casa #####
2022-12-30 11:03:43 WARN .....casa *** Error 'local variable 'ok' referenced before assignment' updating HISTORY
2022-12-30 11:03:43 INFO .....casa Task importgmt complete. Start time: 2022-12-30 16:31:52.733337 End time: 2022-12-30 16:33:43.034807
2022-12-30 11:03:43 INFO .....casa ##### End Task: importgmt #####
2022-12-30 11:03:43 INFO .....casa #####

Insert Message:
fitsfile = 'SGRB-DATA.FITS' # Name of input UV FITS file
flagfile = '' # List of files containing flagging information.
vis = 'multi.ms' # Name of input visibility file

CASA <5>: go importgmt
....10....20....30....40....50....60....70....80....90....100%
2022-12-30 11:03:42 WARN MSFitsInput::fillAntennaTable XYZHAND keyword not found in AN table. Will assume antenna coordinate system is right handed.
2022-12-30 11:03:43 WARN importuvfits::casa *** Error 'local variable 'ok' referenced before assignment' updating HISTORY

```

Fig. 1: Output of the task `importgmt` in the logger and view of the terminal. We will ignore the warnings in the logger for this tutorial.

The output file *multi.ms* file will contain your visibilities in MS format.

An alternative way to run the task is as follows:

```
importgmt(fitsfile='SGRB-DATA.FITS',vis='multi.ms')
```

For the tutorial, we will follow the first method.

The MS format stores the visibility data in the form of a table. The *data* column contains the data. Operations like calibration and deconvolution will produce additional columns such as the *corrected* and *model* data columns.

3.2 Data inspection

You will first want to know about the contents of the MS file. The task `listobs` will provide a summary of the contents of your dataset in the logger window.

```
inp listobs
vis='multi.ms'
go listobs
```

You can also choose to save the output to a text file so that you can refer to it when needed without having to run the task.

```
inp listobs
vis='multi.ms'
listfile='listobs-out.txt'
go listobs
```

Note the scans, field ids, source names, number of channels, total bandwidth, channel width and central frequency for your observations. Field ids (e. g. 0, 1, 2) can be used in subsequent task to choose sources instead of their names (e. g. 3C286, SGRB, etc.).

The task `plotms` is used to plot the data. It opens a GUI in which you can choose to display portions of your data. Go through the help for `plotms` GUI in CASA documentation.

```

2022-12-30 11:17:16 INFO ...casa #####
2022-12-30 11:17:16 INFO ...casa ##### Begin Task: listobs #####
2022-12-30 11:17:16 INFO ...casa listobs(vis='multi.ms', selectdata=True, spw='', field='', antenna='', uvrange='', timerange='', correlation='', scan='', intent
...summary+ #####
2022-12-30 11:17:16 INFO ...summary+ MeasurementSet Name: /data/ruta-casa-test/multi.ms MS Version 2
2022-12-30 11:17:16 INFO ...summary+ #####
2022-12-30 11:17:16 INFO ...summary+ Observer: USER Project:
2022-12-30 11:17:16 INFO ...summary+ Observation: GMRT
2022-12-30 11:17:16 INFO ...summary+ Computing scan and subscan properties...
2022-12-30 11:17:16 INFO ...summary+ Data records: 165300 Total elapsed time = 4853.31 seconds
2022-12-30 11:17:16 INFO ...summary+ Observed from 24-Aug-2019/05:54:49.2 to 24-Aug-2019/07:15:42.5 (TAI)
2022-12-30 11:17:16 INFO ...summary+ #####
2022-12-30 11:17:16 INFO ...summary+ ObservationID = 0 ArrayID = 0
2022-12-30 11:17:16 INFO ...summary+ Date Timerange (TAI) Scan FldId FieldName nRows SpwIds Average Interval(s) ScanIntent
2022-12-30 11:17:16 INFO ...summary+ 24-Aug-2019/05:54:49.2 - 06:18:15.8 1 0 3C286 56985 [0] [10.7]
2022-12-30 11:17:16 INFO ...summary+ 06:29:10.8 - 06:34:00.7 2 1 1248-199 11745 [0] [10.7]
2022-12-30 11:17:16 INFO ...summary+ 06:34:54.4 - 07:09:48.2 3 2 SGRB 84825 [0] [10.7]
2022-12-30 11:17:16 INFO ...summary+ 07:10:52.6 - 07:15:42.5 4 1 1248-199 11745 [0] [10.7]
2022-12-30 11:17:16 INFO ...summary+ (nRows = Total number of rows per scan)
2022-12-30 11:17:16 INFO ...summary+ Fields: 3
2022-12-30 11:17:16 INFO ...summary+ ID Code Name RA Decl Epoch SrcId nRows
2022-12-30 11:17:16 INFO ...summary+ 0 C 3C286 13:31:08.293649 +30.30.32.93706 J2000 0 56985
2022-12-30 11:17:16 INFO ...summary+ 1 C 1248-199 12:48:23.904215 -19.59.18.61580 J2000 1 23490
2022-12-30 11:17:16 INFO ...summary+ 2 C SGRB 13:09:48.084191 -23.22.53.32407 J2000 2 84825
2022-12-30 11:17:16 INFO ...summary+ Spectral Windows: (1 unique spectral windows and 1 unique polarization setups)
2022-12-30 11:17:16 INFO ...summary+ SpwID Name #Chans Frame Ch0(MHz) ChanWid(kHz) TotBW(kHz) CtrFreq(MHz) Corrs
2022-12-30 11:17:16 INFO ...summary+ 0 none 2048 TOPO 550.049 97.656 200000.0 650.0000 RR LL
2022-12-30 11:17:16 INFO ...summary+ Sources: 3
2022-12-30 11:17:16 INFO ...summary+ ID Name SpwID RestFreq(MHz) SysVel(km/s)
2022-12-30 11:17:16 INFO ...summary+ 0 3C286 0 0 1.03605565933e-320
2022-12-30 11:17:16 INFO ...summary+ 1 1248-199 0 0 1.03605565933e-320
2022-12-30 11:17:16 INFO ...summary+ 2 SGRB 0 0 1.03605565933e-320
2022-12-30 11:17:16 INFO ...summary+ Antennas: 30:
2022-12-30 11:17:16 INFO ...summary+ ID Name Station Diam. Long. Lat. Offset from array center (m) ITRF Geocentric coo
2022-12-30 11:17:16 INFO ...summary+ 0 C00 C00:01 45.0 m +074.03.07.6 +18.58.20.6 224.2552 39.5295 620.2676 1657011.579000 5798582.2
2022-12-30 11:17:16 INFO ...summary+ 1 C01 C01:02 45.0 m +074.03.04.0 +18.58.23.6 118.8940 132.7384 286.6627 1657017.879000 5798220.8
2022-12-30 11:17:16 INFO ...summary+ 2 C02 C02:03 45.0 m +074.03.01.3 +18.58.28.2 41.9309 274.1318 -0.4904 1657004.629000 5797894.3
2022-12-30 11:17:16 INFO ...summary+ 3 C03 C03:04 45.0 m +074.02.59.5 +18.58.36.2 -11.2620 521.5699 -309.2437 1656953.429000 5797521.6
2022-12-30 11:17:16 INFO ...summary+ 4 C04 C04:05 45.0 m +074.02.57.7 +18.58.37.8 -64.5556 572.3714 -488.2092 1656953.619000 5797328.4
2022-12-30 11:17:16 INFO ...summary+ 5 C05 C05:06 45.0 m +074.02.59.4 +18.58.19.7 -15.5096 12.6719 1.5542 1657083.749000 5797962.1

```

Fig. 2: Output of the task listobs in the logger.

It is important to make a good choice of parameters to plot, so that you do not end up asking to plot too much data at the same time - this can either lead to crashing of plotms or may take a long time to display the data.

Our aim is to inspect the data for non-working antennas. A good choice would be to limit the fields to calibrators and choosing a single channel and plot Amp Vs Time and iterating over antennas. Another good plot for inspection is to choose a single antenna, choose all the channels and plot Amp Vs Channel while iterating over baselines.

It is good to set the inputs for a task to default before running it.

```
default(plotms)
plotms
```

3.3 Flagging

Editing out bad data (e. g. non-working antennas, RFI affected channels, etc.) is termed as flagging. In our MS file, the bad data will be marked with flags and not actually removed as such - thus the term *flagging*. The task `flagdata` will be used to flag the data. See the detailed CASA documentation on flagging using the task `flagdata`.

Here some typical steps of flagging are outlined to get you started.

Usually the first spectral channel is saturated. Thus it is a good idea to flag the first spectral channel. The input `'spw = 0:0'` sets the choice of spectral window 0, channel number 0.

```
default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'manual'
spw = '0:0'
savepars = True
cmdreason = 'badchan'
go flagdata
```

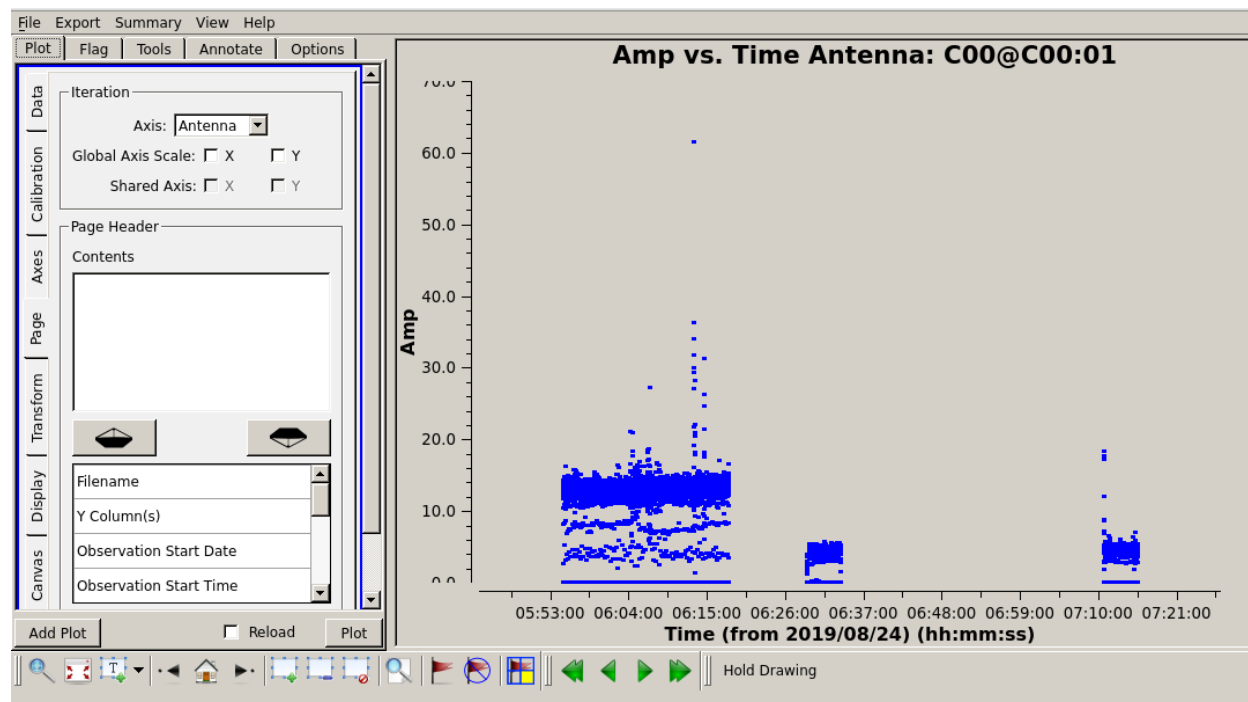


Fig. 3: Screenshot of plotms. Fields 0 and 1 for the channel 400 and correlation rr are plotted for antenna C00. Iteration over antennas in the Page tab seen on the left of the plotms window.

Flagging the first and last records of all the scans is also similarly a good idea. Remember to set the task inputs to *default* before entering new inputs.

```
default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'quack'
quackmode = 'beg'
quackinterval = 10
savepars = True
cmdreason = 'quackbeg'
go flagdata
```

```
default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'quack'
quackmode = 'endb'
quackinterval = 10
savepars = True
cmdreason = 'quackend'
go flagdata
```

In the next step we would like to flag data on antennas that were not working. Using plotms, find out which antennas were not working. Non-working antennas *generally* show up as those having very small amplitude even on bright calibrators, show no relative change of amplitude for calibrators and target sources and the phases towards calibrator sources on any given baseline will be randomly distributed between -180 to 180 degrees. If such antennas are found

in the data, those can be flagged using the task `flagdata`.

Note

Only an example is provided here - you need to locate the bad antennas in the tutorial data and flag those.

Remember also that the some antennas may not be bad at all times. However if an antennas stops working while on the target source, it can be difficult to find out. Thus make a decision based on the secondary calibrator scans. Depending on when such antennas stopped working, you can choose to flag them for that duration. Check the two polarizations separately.

Although `plotms` provides options for flagging data interactively, at this stage, we will choose to just locate the bad data and flag it using the task `flagdata`.

```
default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'manual'
antenna = 'E02, S02, W06'
savepars = True
cmdreason = 'badant'
inp flagdata
go flagdata
```

It is a good idea to review the inputs to the task using (`inp flagdata`) before running it.

Radio Frequency Interferences (RFI) are the manmade radio band signals that enter the data and are unwanted. Signals such as those produced by satellites, aircraft communications are confined to narrow bands in the frequency and will appear as frequency channels that have very high amplitudes. It is not easy to remove the RFI from such channels and recover our astronomical signal. Thus we will flag the affected channels (may be individual or groups of channels). There are many ways to flag RFI - could be done manually after inspecting the spectra or using automated flaggers that look for outliers.

At this stage this will be done on the time-frequency plane for each baseline using the mode `tfcrop` in the task `flagdata`. In the first step we get an idea about the amount of flagging that will result from our choice of parameters `action = 'calculate'` is the parameter that allows us to see this. Since we are still looking at uncalibrated data, we should be flagging only the worst RFI - the flagging percentage of a few percent is ok but if it shows high amount of flagging, one would like to increase the cutoffs used and check again. You may even decide to not use this automated flagger and examine the data further using `plotms` and locate the bad data and flag using `manual` mode in `flagdata`.

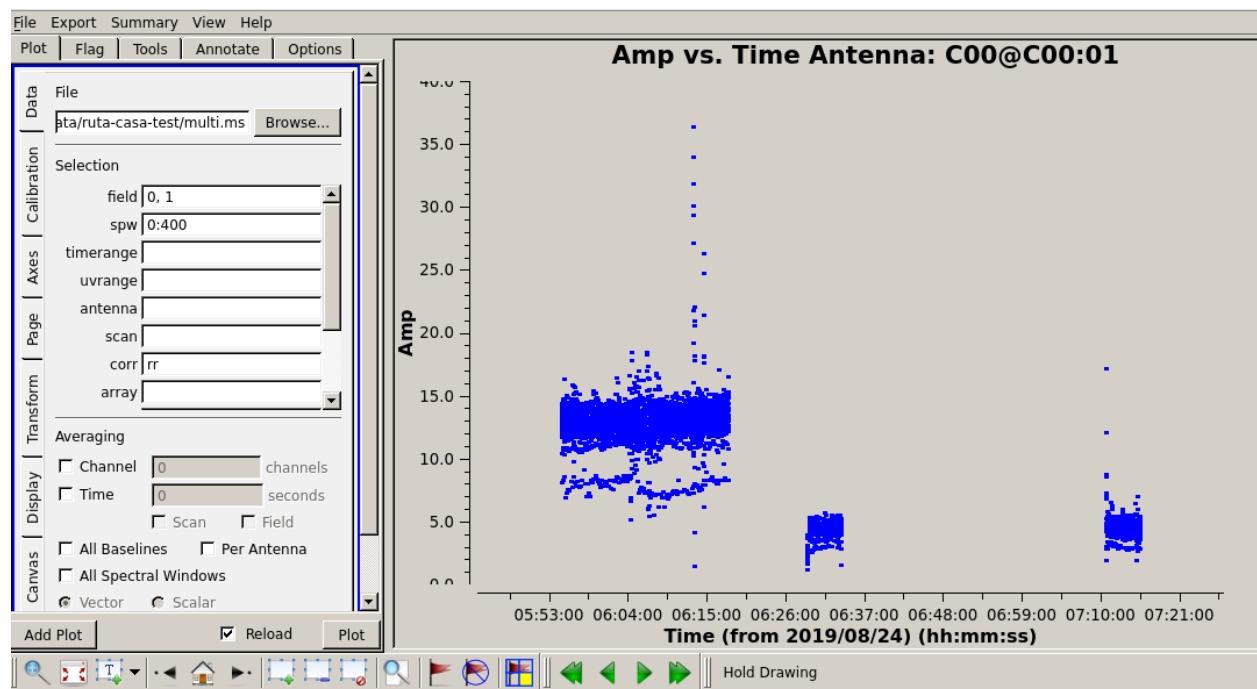
```
default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'tfcrop'
spw = ''
ntime = 'scan'
timecutoff = 6.0
freqcutoff = 6.0
extendflags = False
action = 'calculate'
display = 'both'
inp flagdata
go flagdata
```

If we are satisfied, we could run the same task with `action = 'apply'`.

```

default(flagdata)
inp flagdata
vis = 'multi.ms'
mode = 'tfcrop'
spw = ''
ntime = 'scan'
timecutoff = 6.0
freqcutoff = 6.0
extendflags = False
action = 'apply'
display = ''
inp flagdata
go flagdata

```



Note

If you happen to wrongly flag and would like to restore the older flags, use the task `flagmanager` with `mode = 'list'` to see the flagbackup versions. Locate the version to which you want to restore and run `flagmanager` with `mode = 'restore'` providing the versionname. After that use the task `flagmanager` again with the to delete the unwanted flagbackup versions using it with `mode = 'delete'` and giving the unwanted versionnames.

Now we extend the flags (growtime 80 means if more than 80% is flagged then fully flag, change if required)

```

default(flagdata)
vis = 'multi.ms'
mode = 'extend'
growtime=80.0
growfreq=80.0
action='apply'

```

(continues on next page)

(continued from previous page)

```
inp flagdata
go flagdata
```

3.4 Absolute flux density calibration

In this step, flux densities are set for the standard flux calibrator in the data. The standard flux calibrators used at the GMRT are 3C286, 3C48 and 3C147. We can choose the flux density standard in this task. The default choice is Perley-Butler 2017.

```
default(setjy)
vis = 'multi.ms'
field = '3C147'
go setjy
```

```
2023-01-03 08:47:41 INFO .....casa ##### Begin Task: setjy #####
2023-01-03 08:47:41 INFO .....casa setjy( vis='multi.ms', field='3C286', spw='', selectdata=False, timerange='', scan='', intent='', observation='', scalebychan=True, standa
2023-01-03 08:47:41 INFO .....casa {'field': '3C286'})
2023-01-03 08:47:41 INFO .....open() Opening MeasurementSet /data/ruta-casa-test/multi.ms
2023-01-03 08:47:41 INFO .....setjy() Using channel dependent flux densities
2023-01-03 08:47:41 INFO .....selection Selected 56985 out of 165300 rows.
2023-01-03 08:47:41 INFO .....setjy() 3C286 (fld ind 0) spw 0 [I=22.509, Q=0, U=0, V=0] Jy @ 5.5005e+08Hz, (Perley-Butler 2017)
2023-01-03 08:47:41 INFO .....setjy() Will clear any existing model with matching field=3C286 and spw=
2023-01-03 08:47:41 INFO .....clearing model records in MS header for selected fields.
2023-01-03 08:47:41 INFO .....clearing model records in MS header for selected fields.
2023-01-03 08:47:41 INFO .....selection Selected 56985 out of 165300 rows.
2023-01-03 08:47:41 INFO .....r:ft() Fourier transforming: replacing visibility model header
2023-01-03 08:47:41 INFO .....ation() Processing after subtracting componentlist /data/ruta-casa-test/multi.ms_setjy_spw0_3C286_0.550049GHz58719.3d.cl
2023-01-03 08:47:41 INFO .....chine() Performing interferometric gridding...
2023-01-03 08:47:42 INFO .....casa Task setjy complete. Start time: 2023-01-03 14:17:40.671919 End time: 2023-01-03 14:17:41.638983
2023-01-03 08:47:42 INFO .....casa ##### End Task: setjy #####
2023-01-03 08:47:42 INFO .....casa #####
```

If more than one flux calibrators are present, run it for each of the sources.

3.5 Delay and bandpass calibration

First we will do delay calibration. In calibration, a reference antenna is required. Here “C00” is only taken as an example. You may use any antenna that is working for the whole duration of the observation. We will henceforth be using a central portion of the bandwidth. Depending on the shape of the band, you may change this selection.

```
default(gaincal)
vis = 'multi.ms'
caltable = 'multi.K1'
field = '3C286'
spw = '0:51~1950'
solint = '60s'
refant = 'C00'
solnorm = True
gaintype = 'K'
inp gaincal
go gaincal
```

An initial gain calibration will be done.

```
default(gaincal)
vis = 'multi.ms'
caltable = 'multi.G0'
field = '3C286'
spw = '0:51~1950'
```

(continues on next page)

(continued from previous page)

```
solint = 'int'
refant = 'C00'
minsnr = 2.0
gaintype = 'G'
gaintable = ['multi.K1']
inp gaincal
go gaincal
```

Bandpass calibration using the flux calibrator.

```
default(bandpass)
vis = 'multi.ms'
caltable = 'multi.B1'
field = '3C286'
spw = '0:51~1950'
solint = 'inf'
refant = 'C00'
minsnr = 2.0
solnorm = True
gaintable = ['multi.K1', 'multi.G0']
inp bandpass
go bandpass
```

Examine the bandpass table using plotms. Choose the bandpass table multi.B1 in data and check the plots Amp Vs Channels and Phase Vs Channels iterated over antennas.

Note the shape of the band across the frequencies.

3.6 Gain calibration

A final gain calibration will be done in this step. The task gaincal will be run on the calibrators (flux calibrator/s, and gain calibrator/s).

```
default(gaincal)
vis = 'multi.ms'
caltable = 'multi.AP.G1'
field = '3C286'
spw = '0:51~1950'
solint = '120s'
refant = 'C00'
minsnr = 2.0
gaintype = 'G'
gaintable = ['multi.K1', 'multi.B1']
interp = ['nearest,nearestflag', 'nearest,nearestflag']\
inp gaincal
go gaincal

default(gaincal)
vis = 'multi.ms'
caltable = 'multi.AP.G1'
field = '1248-199'
```

(continues on next page)

(continued from previous page)

```

spw = '0:51~1950'
solint = '120s'
refant = 'C00'
minsnr = 2.0
gaintype = 'G'
gaintable = ['multi.K1', 'multi.B1']
interp = ['nearest,nearestflag', 'nearest,nearestflag']
append = True
inp gaincal
go gaincal

```

The flux density of the phase calibrator will be set in the following step.

```

default(fluxscale)
inp fluxscale
vis = 'multi.ms'
caltable = 'multi.AP.G1'
fluxtable = 'multi.fluxscale'
reference = '3C286'
transfer = '1248-199'
inp fluxscale
go fluxscale

```

```

.....casa ##### Begin Task: fluxscale #####
.....casa fluxscale( vis='multi.ms', caltable='multi.AP.G1', fluxtable='multi.fluxscale', reference=['3C286'], transfer=['1248-199'], listfile='', a
.....er::open ****Using NEW VI2-driven calibrator tool****
.....er::open Opening MS: multi.ms for calibration.
.....brater:: Initializing nominal selection to the whole MS.
.....fluxscale Beginning fluxscale--(MSSelection version)-----
.....cale::: Found reference field(s): 3C286
.....cale::: Found transfer field(s): 1248-199
.....cale::: Flux density for 1248-199 in SpW=0 (freq=6.47754e+08 Hz) is: 7.15158 +/- 0.0397817 (SNR = 179.77, N = 52)
.....fluxscale Storing result in multi.fluxscale
.....cale::: Writing solutions to table: multi.fluxscale
.....casa Task fluxscale complete. Start time: 2023-01-04 12:50:20.477136 End time: 2023-01-04 12:50:21.782205
.....casa ##### End Task: fluxscale #####
.....casa #####

```

3.7 Transfer of gain calibration to the target

First we apply the calibration to the amplitude and phase calibrators. This task creates the *corrected* data column in the MS file.

```

default(applycal)
inp applycal
vis='multi.ms'
field='3C286'
spw = '0:51~1950'
gaintable=['multi.fluxscale', 'multi.K1', 'multi.B1']
gainfield=['3C286','','']
interp=['nearest','','']
calwt=[False]
inp applycal
go applycal

```

```

default(applycal)
inp applycal

```

(continues on next page)

(continued from previous page)

```

vis='multi.ms'
field='1248-199'
spw = '0:51~1950'
gaintable=['multi.fluxscale', 'multi.K1', 'multi.B1']
gainfield=['1248-199','','']
interp=['nearest','','']
calwt=[False]
inp applycal
go applycal

```

Apply calibration to the target.

```

default(applycal)
inp applycal
vis='multi.ms'
field='SGRB'
spw = '0:51~1950'
gaintable=['multi.fluxscale', 'multi.K1', 'multi.B1']
gainfield=['1248-199','','']
interp=['nearest','','']
calwt=[False]
inp applycal
go applycal

```

Use the task `plotms` to examine the calibrated data. You need to select *corrected* data column for plotting.

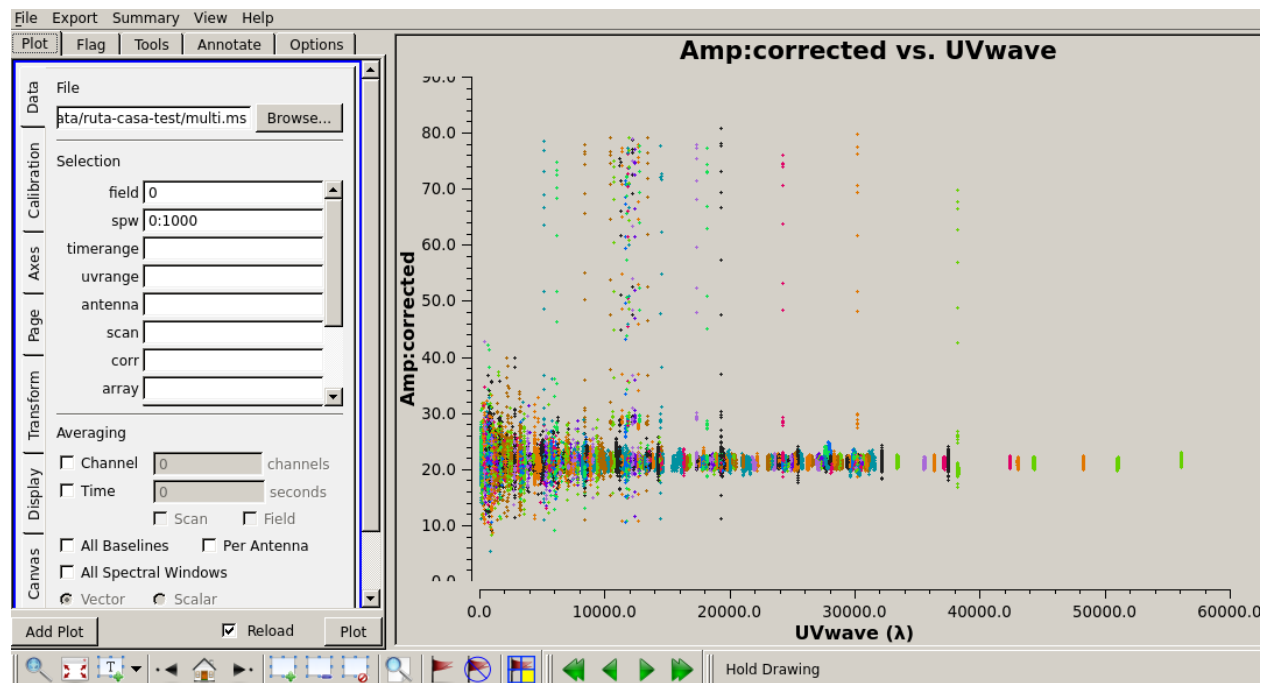


Fig. 4: The calibrated Amp Vs UVwave for the flux calibrator.

You will notice that there are some baselines showing higher amplitudes than the majority or some showing a large scatter.

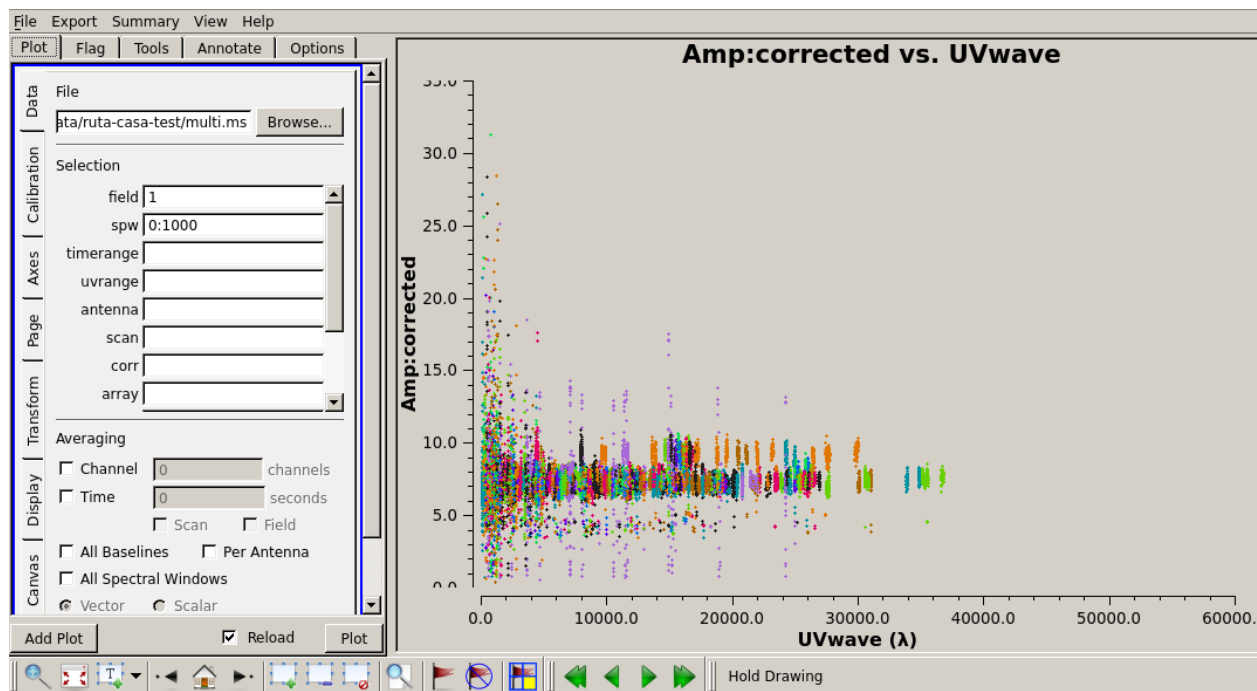


Fig. 5: The calibrated Amp Vs UVwave for the calibrator 1248-199.

Use `plotms` to locate which antennas and baselines have the outlier data. Use the task `flagdata` to flag those data.

3.8 Splitting the calibrated target source data

We will split the calibrated target source data to a new file and do the subsequent analysis on that file. Here we have chosen to drop some more of the edge channels.

```
default(mstransform)
inp mstransform
vis='multi.ms'
outputvis = 'SGB-split.ms'
field='SGB'
spw='0:201~1800'
datacolumn='corrected'
inp mstransform
go mstransform
```

3.9 Flagging on calibrated data

In this section flagging is done on calibrated target source data.

```
default(flagdata)
inp flagdata
vis = 'SGRB-split.ms'
mode = 'tfcrop'
timecutoff = 5.0
freqcutoff = 5.0
timefit = 'poly'
freqfit = 'line'
extendflags=False
inp flagdata
go flagdata
```

You may try out using window statistics and vary ntime to attain better flagging. Though always be careful about not overdoing the flags.

The mode rflag needs to be used with more caution. It is better to select uvranges where you expect the amplitudes to be comparable.

```
default(flagdata)
inp flagdata
vis='SGRB-split.ms'
mode="rflag"
uvrange = '>3klamda'
timedevscale=5.0
freqdevscale=5.0
spectralmax=500.0
extendflags=False
inp flagdata
go flagdata
```

Examine the data using plotms. Flag individual baselines as may be needed.

3.10 Averaging in frequency

The data are averaged in frequency to reduce the volume of the data. However the averaging is done only such that one is not affected by *bandwidth smearing*. For Band 4 it is recommended to average up to 10 channels (when BW is 200 MHz over 2048 channels). However in order to keep the computation time reasonable, for this tutorial we will average 20 channels. You can check the effect of this on the shapes of the sources at the edge of the field.

```
default(mstransform)
inp mstransform
vis='SGRB-split.ms'
outputvis='SGRB-avg-split.ms'
chanaverage=True
chanbin=20
inp mstransform
go mstransform
```

Some more flagging will be done on the data at this stage to get the final dataset for imaging. Use `plotms` to decide on where and what you would like to flag and proceed accordingly.

3.11 Imaging

We will use the task `tclean` for imaging. Go through the CASA documentation for `tclean` to understand the meaning of the various parameters. To save on computation, in this example we have set `wprojplanes=128`. In general to account for the w-term more accurately set `wprojplanes=-1` so that it will be calculated internally in CASA. In this example we have used `interactive = False` in `tclean`.

```
default(tclean)
inp tclean
vis='SGRB-avg-split.ms'
imagename='SGRB-img'
imsize=7200
cell='1.0arcsec'
specmode='mfs'
gridder='wproject'
wprojplanes=128
pblimit=-0.0001
deconvolver='mtmfs'
nterms=2
weighting='briggs'
robust=0
niter=2000
threshold='0.01mJy'
cyclefactor = 0.5
interactive=False
usemask='auto-multithresh'
sidelobethreshold=2.0
pbmask=0.0
savemodel='modelcolumn'
inp tclean
go tclean
```

Keep an eye on the logger messages to get an idea about the processing and in particular see the total flux cleaned. Once the task ends successfully, you can view the image using `imview`. The main image you would like to see is the one with the extension `image.tt0`. Adjust the data range to display it in a reasonable colour scale to show the peaks as well as the noise in the image. Also inspect the rest of the images created by `tclean` - the `.psf`, `.mask`, `.model`, `.residual`.

```
imview
```

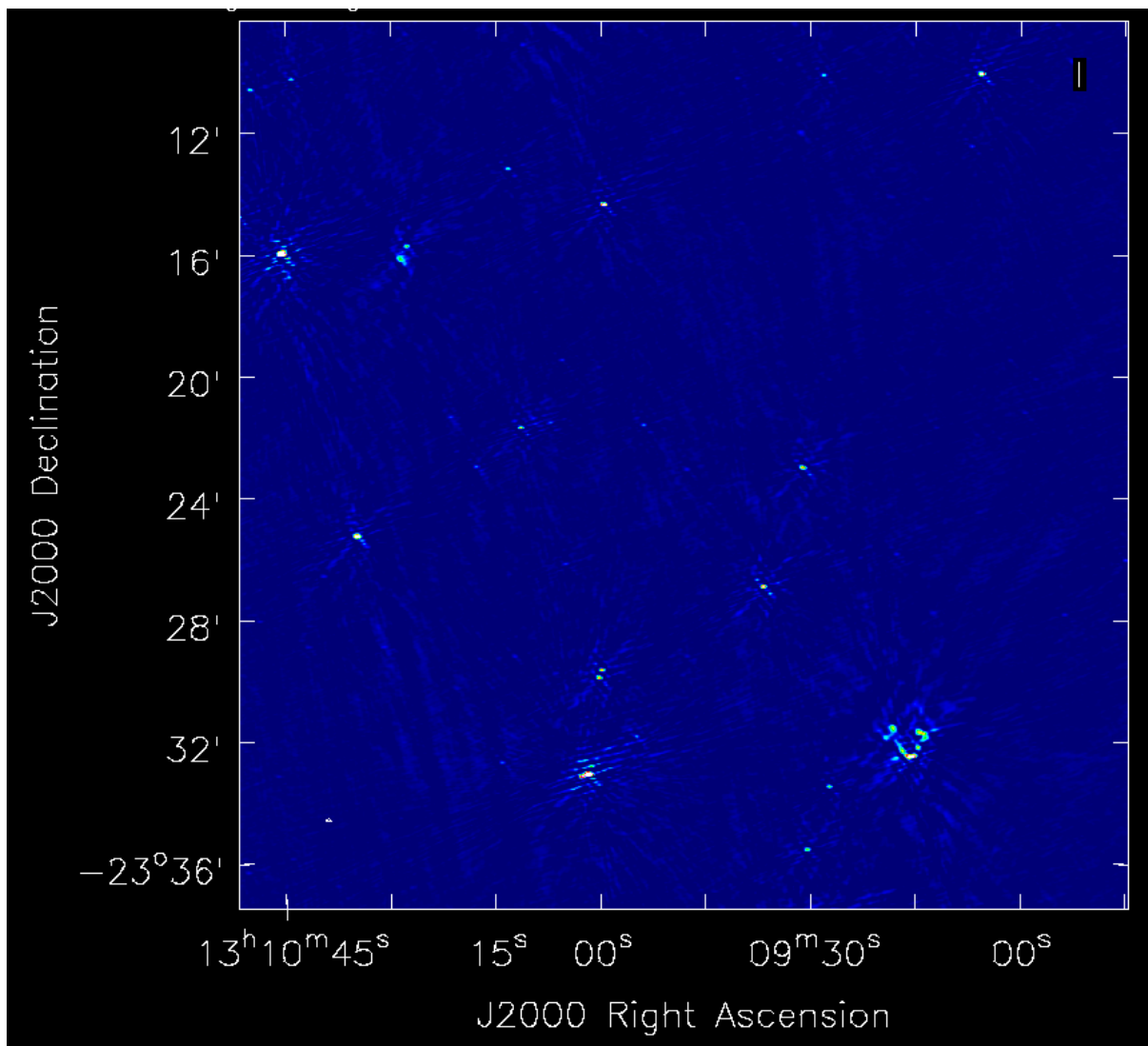


Fig. 6: A zoom-in on the central region of the first image.

3.12 Self-calibration

This is an iterative process. The model from the first tclean is used to calibrate the data and the corrected data are then imaged to make a better model and the process is repeated. The order of the tasks is tclean, gaincal, applycal, tclean. A reasonable choice is to do 5 phase only and two amplitude and phase self-calibrations. We start from a longer “solint” (solution interval), for e. g. “8min” and gradually lower it to “1min” during the phase only iterations. For “a&p” self-calibration, again choose a longer solint such as “2min”. We keep solnorm=False in phase only iterations and solnorm=True in “a&p” self-calibration. As the iterations advance, the model sky is expected to get better so in the task tclean, lower the threshold and increase niter.

```
default(gaincal)
vis = 'GRB-avg-split.ms'
caltable='selcal-p1.GT'
solint = '8min'
refant = 'C00'
minsnr = 3.0
gaintype = 'G'
solnorm= False
calmode = 'p'
go gaincal
```

Using the task “plotcal” you can examine the gain table. In successive iterations of self-calibration, you should find that the phases are more and more tightly scattered around zero. If this trend is not there, you can suspect that the self-calibration is not going well - check the previous tclean runs to see if the total cleaned flux was increasing.

```
default(applycal)
vis='SGRB-avg-split.ms'
gaintable='selcal-p1.GT'
applymode='calflag'
interp=['linear']
go applycal
```

We will split the corrected data to a new file and use it for imaging. This is just for better book-keeping. You may choose to do the successive self-calibration iterations on the same MS file but remember that the corrected data and model data columns will be overwritten by applycal and tclean.

```
default(mstransform)
vis='SGRB-avg-split.ms'
datacolumn='corrected'
outputvis='vis-selfcal-p1.ms'
go mstransform
```

In the next iteration we will use a larger niter and lower the threshold. In the tclean messages do check the total cleaned flux and the number of iterations needed to reach that.

```
default(tclean)
imagename='SGRB-img1'
imsize=7200
cell='1.0arcsec'
specmode='mfs'
gridder='wproject'
```

(continues on next page)

(continued from previous page)

```
wprojplanes=128
pblimit=-0.0001
deconvolver='mtmfs'
nterms=2
weighting='briggs'
robust=0
niter=4000
threshold='0.01mJy'
cyclefactor = 0.5
interactive=False
usemask='auto-multithresh'
sidelobethreshold=2.0
pbmask=0.0
savemodel='modelcolumn'
inp tclean
go tclean
```

Repeat until you stop seeing improvement in the image sensitivity.

After you get your final image you need to do a primary beam correction. The task “widebandpbcor” in CASA does not have the information of the GMRT primary beam shape. A modified version of this task called [ugmrtpb](#) has been written for the uGMRT primary beam correction. You can follow the instructions there to do a primary beam correction for your image.

For this tutorial on the RAS machines please see the steps [here](#).

Acknowledgements: We thank Ishwara Chandra who provided the data used in the Radio Astronomy School for the CASA tutorial. We also thank Nissim Kanekar and Ruta Kale who helped make the first version of this tutorial. The original tutorial has been converted to html by Ruta Kale with help from Shilkumar Meshram.

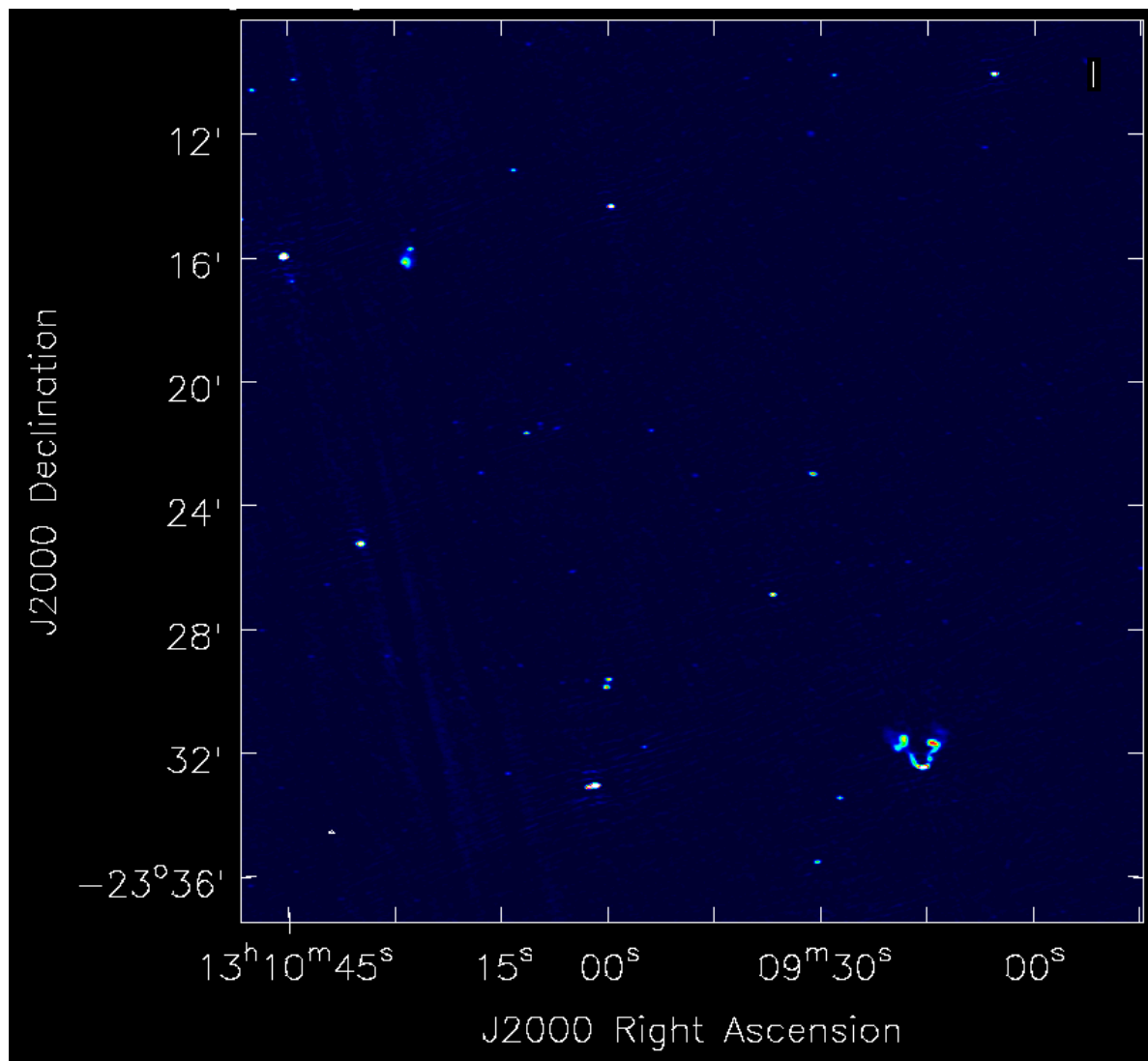


Fig. 7: A zoom-in on the central region of the image after self-calibration.

SPECTRAL LINE DATA REDUCTION USING CASA

For this tutorial you will need to use CASA version 5.6.0. The tar ball for this version is available in `/home/ras23/CASA-tutorials` folder.

You can find the tutorial for reducing spectral line data over [here](#).

The new tasks that you need to use are available in the github [repository](#) by Aditya Chowdhury.

Acknowledgement: We thank Nissim Kanekar, Balpreet Kaur and Aditya Chowdhury for preparing the tutorial.

PULSAR DATA ANALYSIS

The tutorial is intended to provide you a basic introduction to the steps involved in the analysis of pulsar data, including searching for new pulsars/transients and timing a newly discovered pulsar. The filterbank data obtained from GMRT are converted to SIGPROC filterbank format using either the `filterbank` command from SIGPROC or the `rficlean` command from RFIClean. The tutorial will use data already converted to the SIGPROC filterbank format.

5.1 Introduction

You need to have PRESTO, SIGPROC, TEMPO2, TEMPO, and their dependencies installed on your machine. You also need the SIGPROC filterbank data to be available on your disk.

The header information of the data from a SIGPROC filterbank file can be inspected using the `readfile` command from PRESTO (an example using the `header` command from SIGPROC will be shown later).

```
readfile inputFile.fil
```

5.1.1 Data-inspection

A section of the raw data can be inspected by plotting the data as a function of time and frequency, e.g., using the `waterfall.py` command from PRESTO. The command `waterfall.py` has several provisions that enable inspecting the data in several ways (e.g., before and after dedispersion, partial and full averaging over the bandwidth, partial averaging over time, etc.)

```
waterfall.py inputFile.fil -d 0 --subdm 0 --show-ts -s 64 -T 0 -t 5
```

The single pulses from a reasonably strong pulsars might be visible after dedispersing the data at correct dispersion measure (DM).

```
waterfall.py inputFile.fil -d <DM> --subdm <DM> --show-ts -s 64 -T 0 -t 5
```

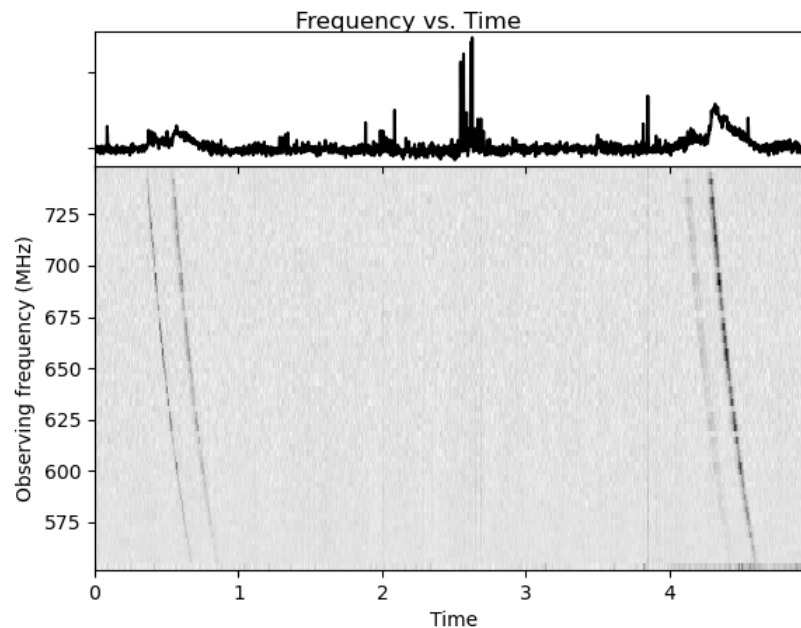
Assuming the data is a SIGPROC filterbank file.

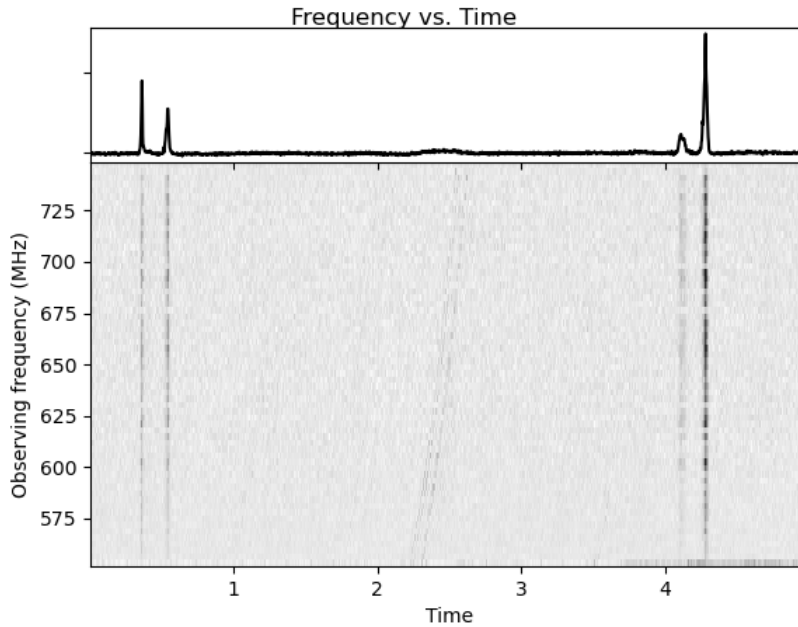
1: From the SIGPROC filterbank file 'introTestd4.fil':

```

    Telescope = GMRT
    Source Name = B0525+21
    Obs Date String = 2019-10-31T21:33:53.1809
    MJD start time = 58787.89853218595817
    RA J2000 = 05:28:52.3000
    RA J2000 (deg) = 82.21791666666667
    Dec J2000 = 22:00:04.0000
    Dec J2000 (deg) = 22.001111111111111
    Tracking? = True
    Azimuth (deg) = 55.47805
    Zenith Ang (deg) = 5.455515
    Number of polns = 2 (summed)
    Sample time (us) = 2621.44
    Central freq (MHz) = 650.09765625
    Low channel (MHz) = 550.1953125
    High channel (MHz) = 750
    Channel width (MHz) = 0.1953125
    Number of channels = 1024
    Total Bandwidth (MHz) = 200
    Beam = 1 of 1
    Beam FWHM (deg) = 1.000
    Spectra per subint = 2400
    Spectra per file = 143051
    Time per subint (sec) = 6.291456
    Time per file (sec) = 374.99961344
    bits per sample = 8
    Are bytes signed? = False
    bytes per spectra = 1024
    samples per spectra = 1024
    bytes per subint = 2457600
    samples per subint = 2457600
    zero offset = 0
    Invert the band? = False
    bytes in file header = 315

```





5.1.2 Dedispersion and Folding

A small section of the raw data can be dedispersed and viewed using `waterfall.py` as seen above. To dedisperse the whole data and also fold it over the rotation period of the pulsar, we can use `prepdata` from PRESTO.

```
prepfold -p 3.7452943 -dm 50.9 -topo -n 128 -nosearch introTestd4.fil
```

`prepfold` is a powerful command, it can search for the most optimum parameters around the given fiducial parameters.

images/pulsar/prepout.png

5.2 Searching for new pulsars

This part of the tutorial aims to demonstrate the process of pulsar search using observations from GMRT. We will use band-4 (550-750 MHz) data of a test pulsar.

5.2.1 Introduction

Pulsars are fascinating extraterrestrial objects that produce pulse like emission with very precise periodicity. These objects are often detected in time domain radio observations. In this tutorial, we will learn how to search for these objects using the strengths of individual pulsed emission as well as their precise periodic behaviour. This tutorial will require PRESTO installed on your machine along with a filterbank file with pulsar in it.

5.2.2 Checking filterbank file

Filterbank file is a file-format for beam data from radio telescopes. This file has the metadata of the observation along with the raw beamdata recorded in the telescope. Use following syntax to check the metadata,

```
header <filterbank_file.fil>
```

```
Data file           : J1534-5334_pa_550_200_4096_4_1_8_16jun2021.fil
Header size (bytes) : 390
Data size (bytes)   : 21441286144
Data type           : filterbank (topocentric)
Telescope           : GMRT
Datataking Machine  : GMRTNEW
Source Name         : J1534-5334
Source RA (J2000)   : 15:34:00.0
Source DEC (J2000)  : -53:34:00.0
Frequency of channel 1 (MHz) : 749.942310
Channel bandwidth    (MHz) : -0.048820
Number of channels   : 4096
Number of beams      : 1
Beam number          : 1
Time stamp of first sample (MJD) : 59381.692884735159
Gregorian date (YYYY/MM/DD) : 2021/06/16
Sample time (us)     : 81.92000
Number of samples     : 5234689
Observation length (minutes) : 7.1
Number of bits per sample : 8
Number of IFs         : 1
```

Fig. 1: Example of a filterbank header

5.2.3 Dedispersion

For an unknown pulsar, we don't know the dispersion caused by the interstellar medium. We try a lot of trial DM values and perform search in each dedispersed time series. We will use `prepsubband` to dedisperse the filterbank file at a range of trial DM values,

```
prepsubband -lodm <smallest_trial_dm> -dmstep <seperation_between_dms> -numdms <number_
of_trial_dms> -numout <number_of_output_samples> <input_fil_name> -o <output_base_name>
```

Please note that number of output samples should be an even number (in order to get a straight-forward FFT in later stage). This will create a number of dedispersed time series with names `output_base_name_dm.dat`. These are the time series that will be used for single pulse search as well as periodicity search.

5.2.4 Exploring the time series

Now, since we have the dedispersed time series, we can try to visualize the pulses in the time series. Use `exploredat` to visualize the time series,

```
exploredat output_base_name.dat
```

This will generate an interactive plot,

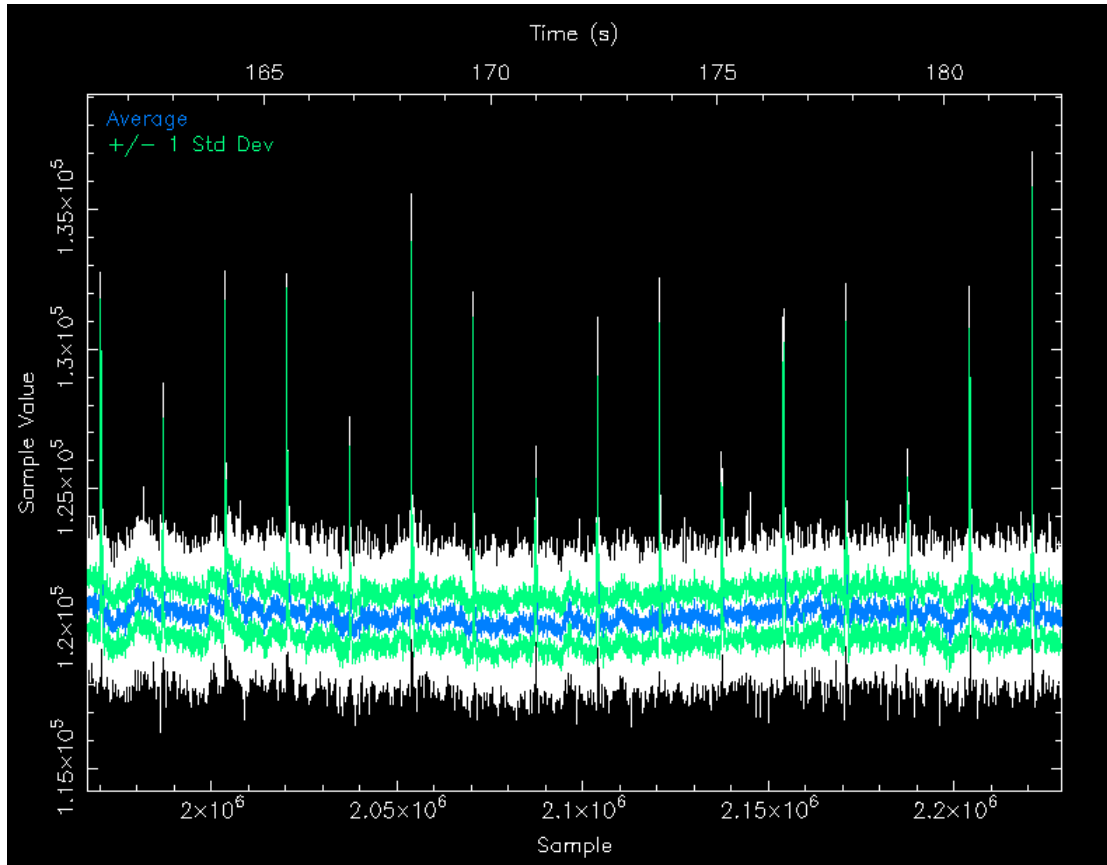


Fig. 2: Output plot of `exploredat`. One can zoom in, zoom out, and move around in this plot.

You can even try to estimate the period of the repeating signal from this plot.

5.2.5 Single pulse search

Though we have seen the single pulses from the pulsar directly in the time series, there is a formal way to search for single pulses in the time series.

```
single_pulse_search.py -t <significance_threshold> -m <maximum_width> <dedispersed_time_
Series.dat>
```

This generates a text file with information of individual detected pulses for each time series. Now, we need to combine the results of each search in a single summary plot,

```
single_pulse_search.py -t <significance_threshold> <_dedispersed_time_Series.singlepulse>
```

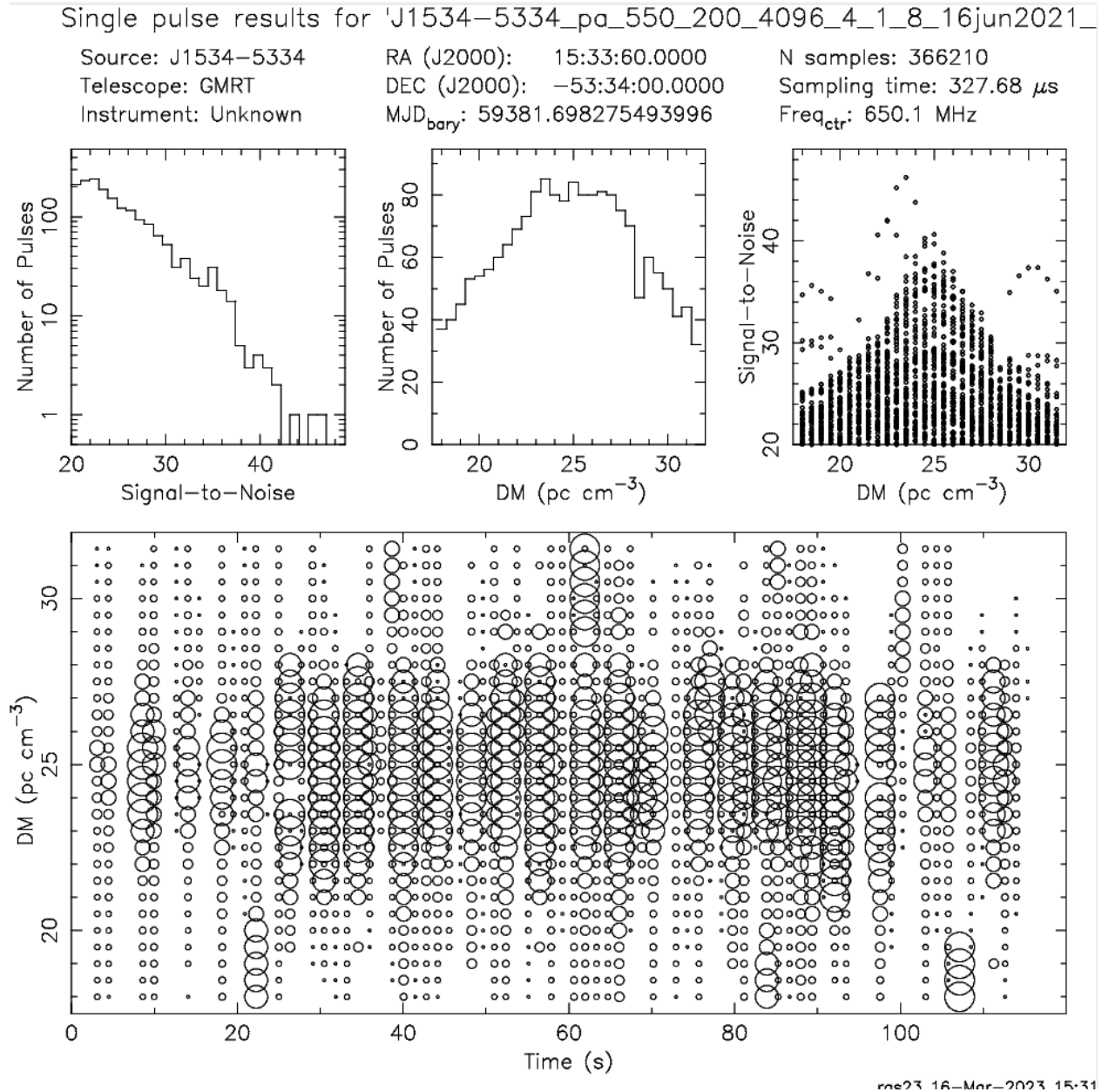
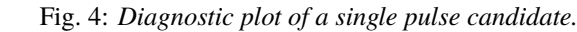


Fig. 3: *Output of single pulse search. We get location of pulses, number of pulses, and single pulse S/Ns as a function of DM.*

You can note down timestamp of some strong pulses from the text file and try to verify it's presence in the time series using `exploredat`. The candidate plot of a single pulse is contains various diagnostic plots,



There are pulsars that are weak and can not be seen in single pulses. To recover such pulsars, we use their periodic property to get a good indication of their presence in the time series. Here, we will see one such method called FFT based periodicity search. The process is to take Fourier transform of the time series and search for strong signal in the power spectra. First we will see how FFT of a time series with repeating signal looks like. To compute the FFT of the time series,

You will see signal at fundamental and subsequent harmonics.

We use `accelsearch` to search for periodic signal, after cleaning the FFT of timeseries using `rednoise`

This will generate a text file containing the results of the search.

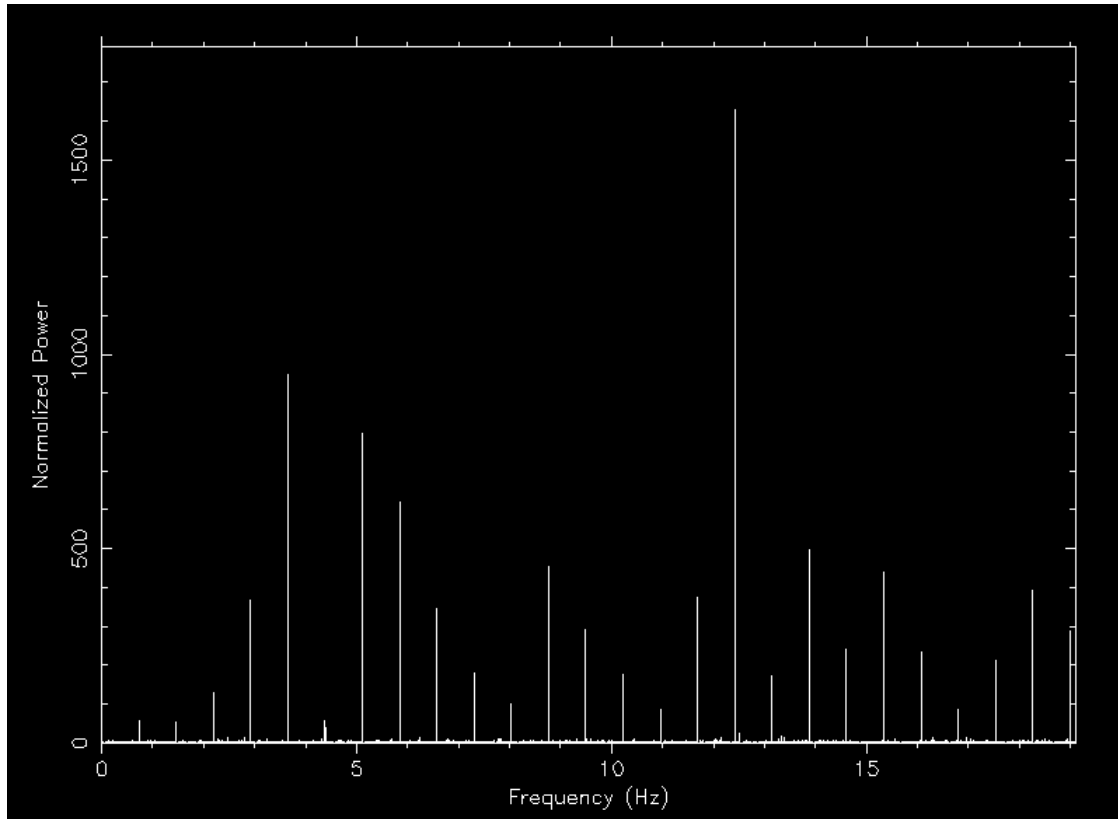


Fig. 6: Visualization of power-spectra of a time series. Strong signal can be seen at the fundamental and subsequent harmonics.

Cand	Sigma	Summed Power	Coherent Power	Num Harm	Period (ms)	Frequency (Hz)	FFT 'r' (bin)	Freq Deriv (Hz/s)	FFT 'z' (bins)	Accel (m/s ²)	Notes
1	199.8	20106.1	119390.3	16	684.42(3)	1.46109(7)	626.500(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(1.4)x10 ⁻²	SL H1 J1502-5828
2	134.0	9104.6	3407.60	16	195.555(3)	5.11366(7)	2192.688(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(4.0)x10 ⁻¹	SL H9 B1454-51
3	121.5	7509.1	3148.76	16	152.099(2)	6.57468(7)	2819.156(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(3.1)x10 ⁻¹	PSR B1509-58
4	105.5	5693.9	3079.76	16	124.445(1)	8.03569(7)	3445.625(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(2.5)x10 ⁻¹	SL H14 B1454-51
5	104.0	5480.3	3626.02	8	105.299(2)	9.4968(1)	4072.125(63)	0.0(1)x10 ⁻⁵	0.00(25)	0.0(4.3)x10 ⁻¹	8th H J1502-6128
6	84.98	3723.7	39.76	16	570.37(2)	1.75326(7)	751.781(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(1.2)x10 ⁻²	SL H3 B1454-51
7	82.25	3494.1	556.05	16	256.664(5)	3.89614(7)	1670.625(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(5.2)x10 ⁻¹	SL H8 J1512-5431
8	78.47	3189.2	64.59	16	365.03(1)	2.73947(7)	1174.656(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(7.4)x10 ⁻¹	SL H6 J1519-6106
9	78.45	3142.8	6707.42	8	72.0466(8)	13.8799(1)	5951.563(63)	0.0(1)x10 ⁻⁵	0.00(25)	0.0(2.9)x10 ⁻¹	SL H30 J1519-6106
10	78.08	3114.2	5735.93	8	80.5230(9)	12.4188(1)	5325.063(63)	0.0(1)x10 ⁻⁵	0.00(25)	0.0(3.3)x10 ⁻¹	13th H J1513-5946
11	71.26	2579.3	5485.65	4	54.7554(9)	18.2630(3)	7831.00(13)	0.0(3)x10 ⁻⁵	0.00(50)	0.0(4.5)x10 ⁻¹	SL H32 B1454-51
12	68.68	2398.7	5343.06	4	59.517(1)	16.8019(3)	7204.50(13)	0.0(3)x10 ⁻⁵	0.00(50)	0.0(4.9)x10 ⁻¹	9th H J1502-5653
13	62.28	2042.9	3832.51	16	479.11(2)	2.08720(7)	894.969(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(9.8)x10 ⁻¹	SL H2 B1507-44
14	62.19	2036.9	1158.98	16	1505.7(2)	0.66415(7)	284.781(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(3.1)x10 ⁻²	PSR J1457-5900
15	59.01	1779.9	3386.97	4	47.2028(6)	21.1852(3)	9084.00(13)	0.0(3)x10 ⁻⁵	0.00(50)	0.0(3.8)x10 ⁻¹	SL H37 B1454-51
16	58.49	1749.4	2125.74	4	44.1573(6)	22.6463(3)	9710.50(13)	0.0(3)x10 ⁻⁵	0.00(50)	0.0(3.6)x10 ⁻¹	SL H34 J1457-5900
17	47.05	1202.0	2963.77	16	213.890(3)	4.67529(7)	2004.719(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(4.4)x10 ⁻¹	7th H J1457-5900
18	41.53	899.14	1850.52	4	36.9969(4)	27.0293(3)	11589.88(13)	0.0(3)x10 ⁻⁵	0.00(50)	0.0(3.0)x10 ⁻¹	SL H24 J1517-4636
19	41.01	932.33	676.27	16	162.962(2)	6.13638(7)	2631.219(31)	0.0(7)x10 ⁻⁶	0.00(13)	0.0(3.3)x10 ⁻¹	SL H6 J1513-5739

Fig. 7: Result of accelsearch. First detection is the first harmonic of the pulsar signal.

5.2.7 Folding filterbank file

Once we know the correct period and DM of the pulsar, we can fold the filterbank file to generate characteristic plots of the pulsar. We use `prepfold` to fold a filterbank,

```
prepfold -p <period> -dm <DM> -nosearch -zerodm <filterbank_file.fil>
```

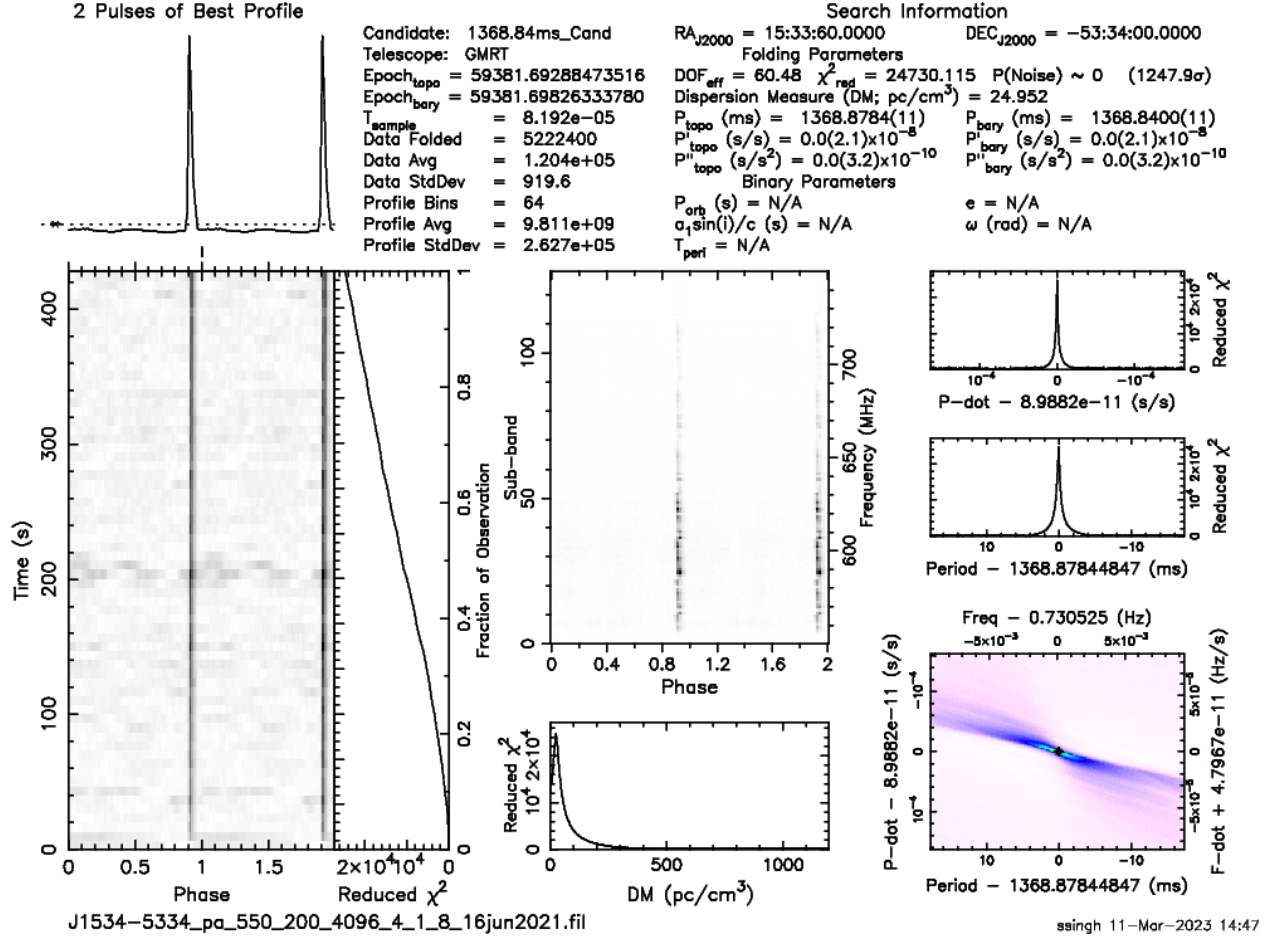


Fig. 8: Result of `prepfold`. Profile of the pulsar along with subintegration vs phase, frequency vs phase, S/N vs DM, S/N vs period plots.

There are lot of such plots for each trial DM in a blind search. All these plots are inspected either by eye or an machine learning based classifier to tell which one is a pulsar and which one is not. Once a pulsar is identified, one needs to check if it's a new pulsar or an already known pulsar.

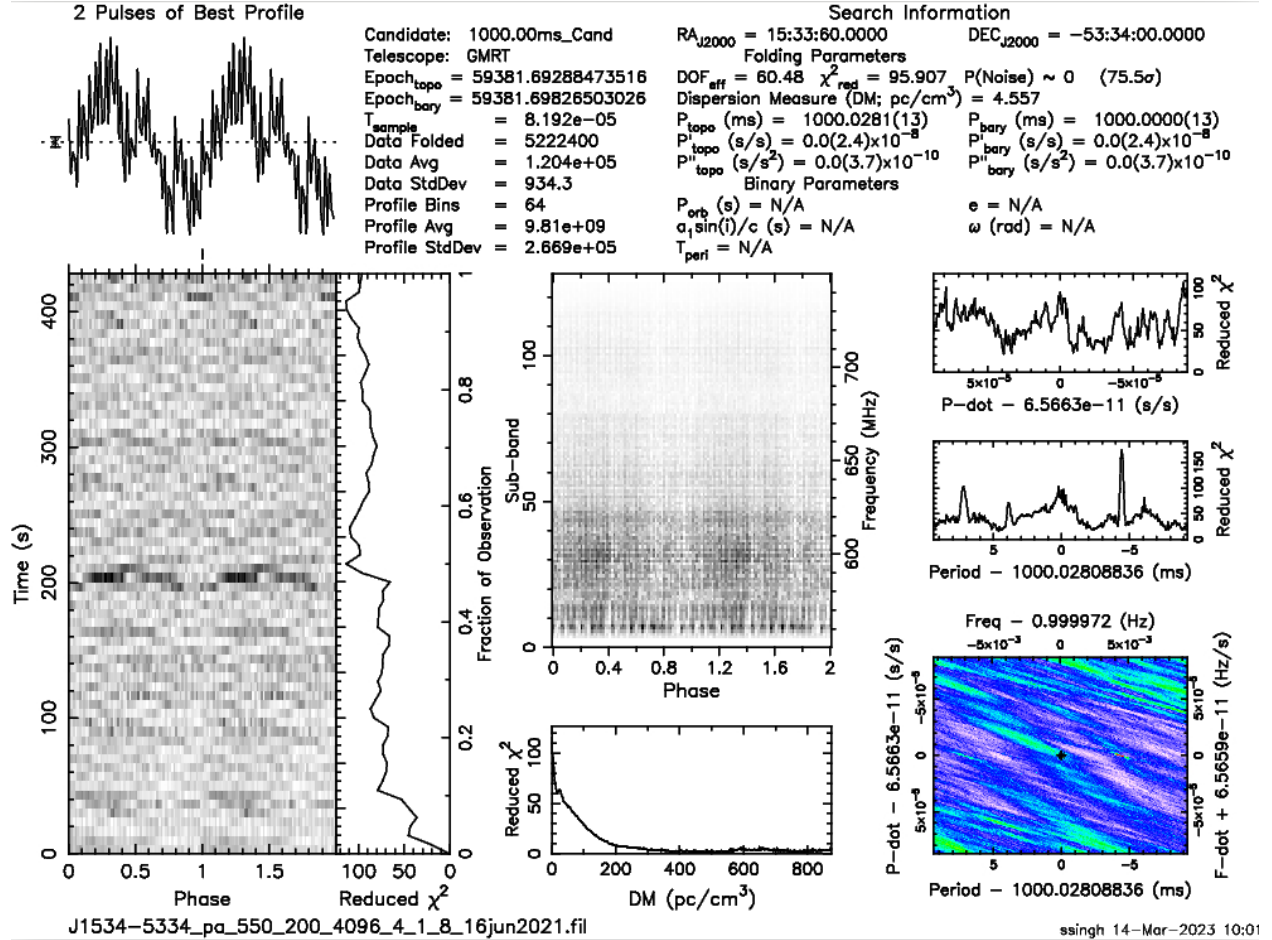


Fig. 9: A false candidate. The pulse seen in the profile is caused by RFI in the time series.

5.3 Timing of Pulsars

The tutorial is intended to provide you an introduction to the basic steps related to the timing analysis of pulsar using GMRT beam data. We will be using the band-3 (300 - 500 MHz) datasets for this purpose. We will only consider the case of an isolated pulsar for the exercise.

We will use PSRCHIVE and TEMPO2 Softwares.

5.3.1 Introduction

The GMRT beam data is in binary format and is converted to “filterbank” format (same binary file with a header). The “filterbank” is folded based on pulsar ephemeris, yielding three-dimensional (time, frequency, phase) folded pulsar data (“pfd”). For this tutorial, we will begin with “pfd” files.

You need to have PSRCHIVE and TEMPO2 installed on your machine. You also need the “pfd” files to be available on your disk.

5.3.2 Prediction of phases of pulsar using TEMPO2

Right Ascension (RA) and Declination (DEC) provide the source’s two-dimensional position in the sky, which is typically defined by the J2000 system. The pulsar’s spin frequency (F0) is defined as the reciprocal of its spinning period. F1 is the first derivative of spin frequency with respect to time. Since the pulsar slows down over time, F1 has a negative value. The dispersion measure (DM) quantifies the line of sight column density of electrons. All of these parameters are listed in a file called the pulsar’s parameter file. Let’s start with a parameter file of pulsar J1646-2142, which contains the pulsar’s name, RA, DEC, F0, F1, and DM values.

```
cat J1646-2142.par
```

```
PSRJ          J1646-2142
RAJ           16:46:18.6346951      1  0.00093156362605884802
DECJ          -21:42:02.41945       1  0.10038077283324266726
POSEPOCH      56596.524252000000001
F0            170.8494057188668898   1  0.00000000023616865636
F1            -2.4104133616727776908e-16 1  1.1566167347732117808e-18
PEPOCH        56596.524252000000001
DM            29.758599393045838389   0.00130264869379864354
TZRSITE       gmrt
START         58418.247898154593713   1
FINISH        59428.51284883502376    1
TZRMJD        58909.904085008769506
TZDF0         287.68712123000002405
```

```
tempo2 -gr fake -f J1646-2142.par -nobsd 1 -ndobs 7 -randha y -ha 8 -start 59000 -end_
↪60000 -rms 1e-3
```

This will generate time of arrivals (ToAs) for this pulsar ranging from MJD 59000 to 60000. ToAs are measured for weekly observations (i.e. one observation per seven days). A single ToA is generated for each observation. Each ToA

will have a 1e-3 millisecond error (i.e. 1 microsecond). The observation hour angle is chosen at random over an 8-hour period.

At the end of this a file with name J1646-2142.simulate is created. The file J1646-2142.simulate contains the predicted ToAs of the pulsar.

```
head J1646-2142.simulate
```

```
FORMAT 1
MODE 1
fake.rf 1440.00000000 59000.82293121699986571 1.00000 7
fake.rf 1440.00000000 59007.50661501823337218 1.00000 7
fake.rf 1440.00000000 59014.74664617557274937 1.00000 7
fake.rf 1440.00000000 59021.73776030585399610 1.00000 7
fake.rf 1440.00000000 59028.79411885326734577 1.00000 7
fake.rf 1440.00000000 59035.29894180085182498 1.00000 7
fake.rf 1440.00000000 59042.37160969282684775 1.00000 7
fake.rf 1440.00000000 59049.64116788072028541 1.00000 7
```

Normally, the first column represents the file name; in this case, it is the same here because it was predicted using tempo2's fake plugin. The frequency is represented in megahertz in the second column. The third and fourth columns contain the ToAs for the observation and the associated error. Note that we maintained the same error during prediction for all observations. The the last column contains the observatory site code.

5.3.3 The impact of parameter errors on timing data

First check the phases of the pulsar are derived accurately by running the command:

```
tempo2 -nofit -f J1646-2142.par J1646-2142.simulate -gr plk
```

The command above displays the difference between the ToAs predicted by the parameter file J1646-2142.par and the ToAs listed in the timing file J1646-2142.simulate. The residuals must be randomly distributed around the zero line if the ToAs listed in the J1646-2142.simulate file matched the prediction from the J1646-2142.par file. The figure below demonstrates this.

Now let's introduce a small error (say around an arc-sec) in position (RA and DEC) in the parameter file to see the effect of wrong position on the timing data. After putting the offset in the parameter file run the above command again which will show that an error in position has yearly periodic variation in the residuals. This is demonstrated in figure below.

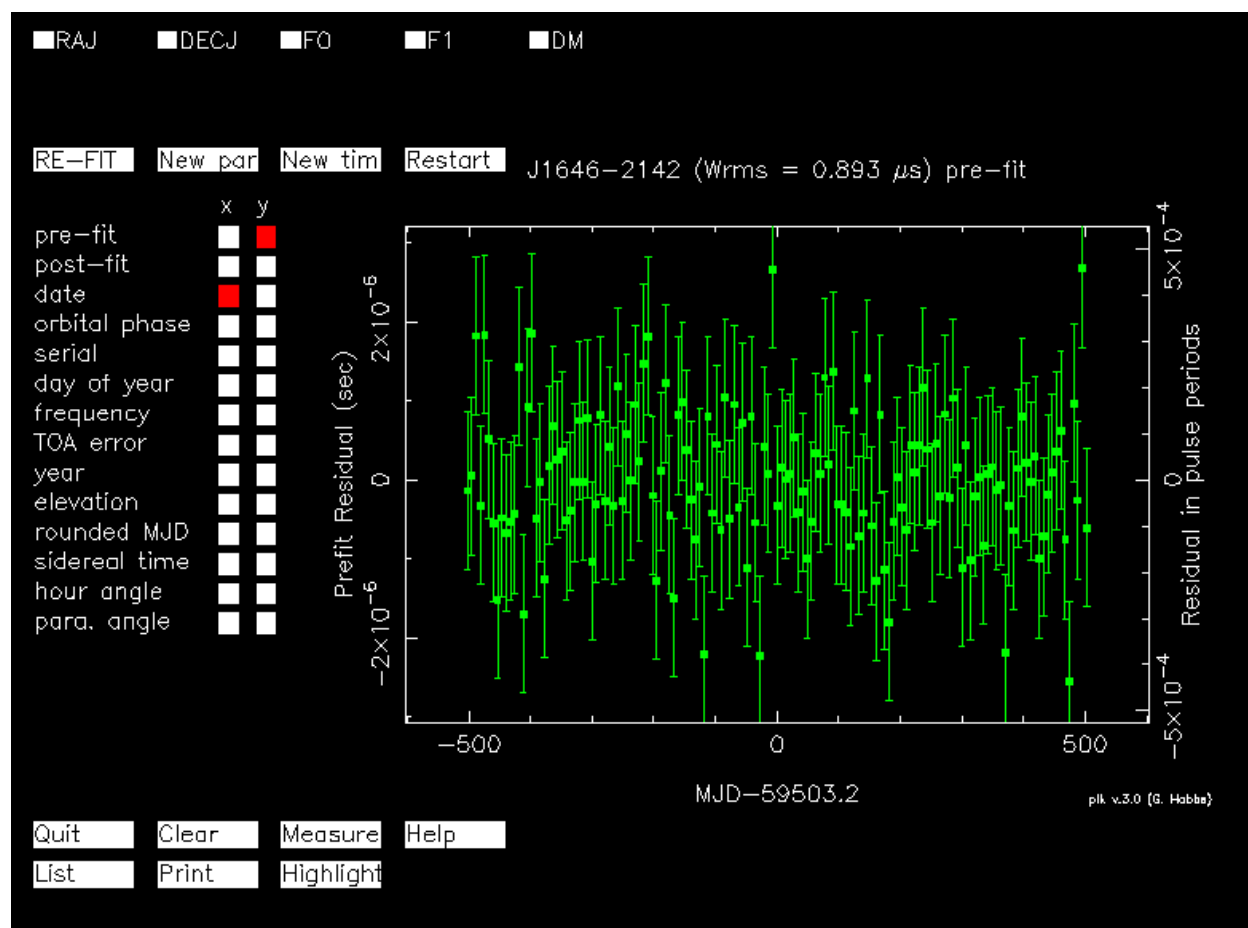
If you fit for the RA/DEC for which the error is introduced, you will again get the randomly sampled residuals around the zero line in the post-fit residuals.

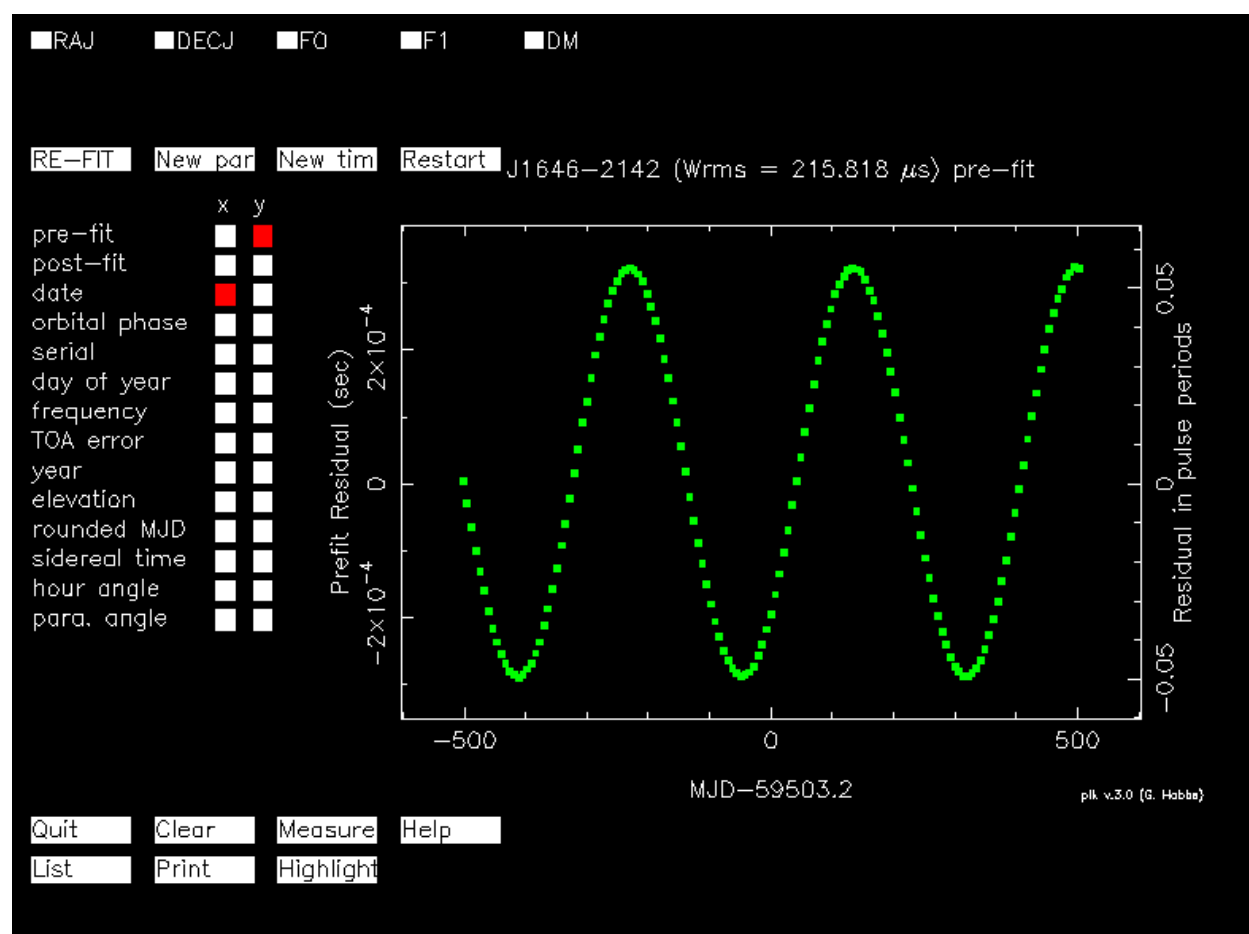
Now, reset the parameter file to its default values and try introducing a small error in F0. The residuals for the F0 error show a straight line pattern with no periodicity, as shown below.

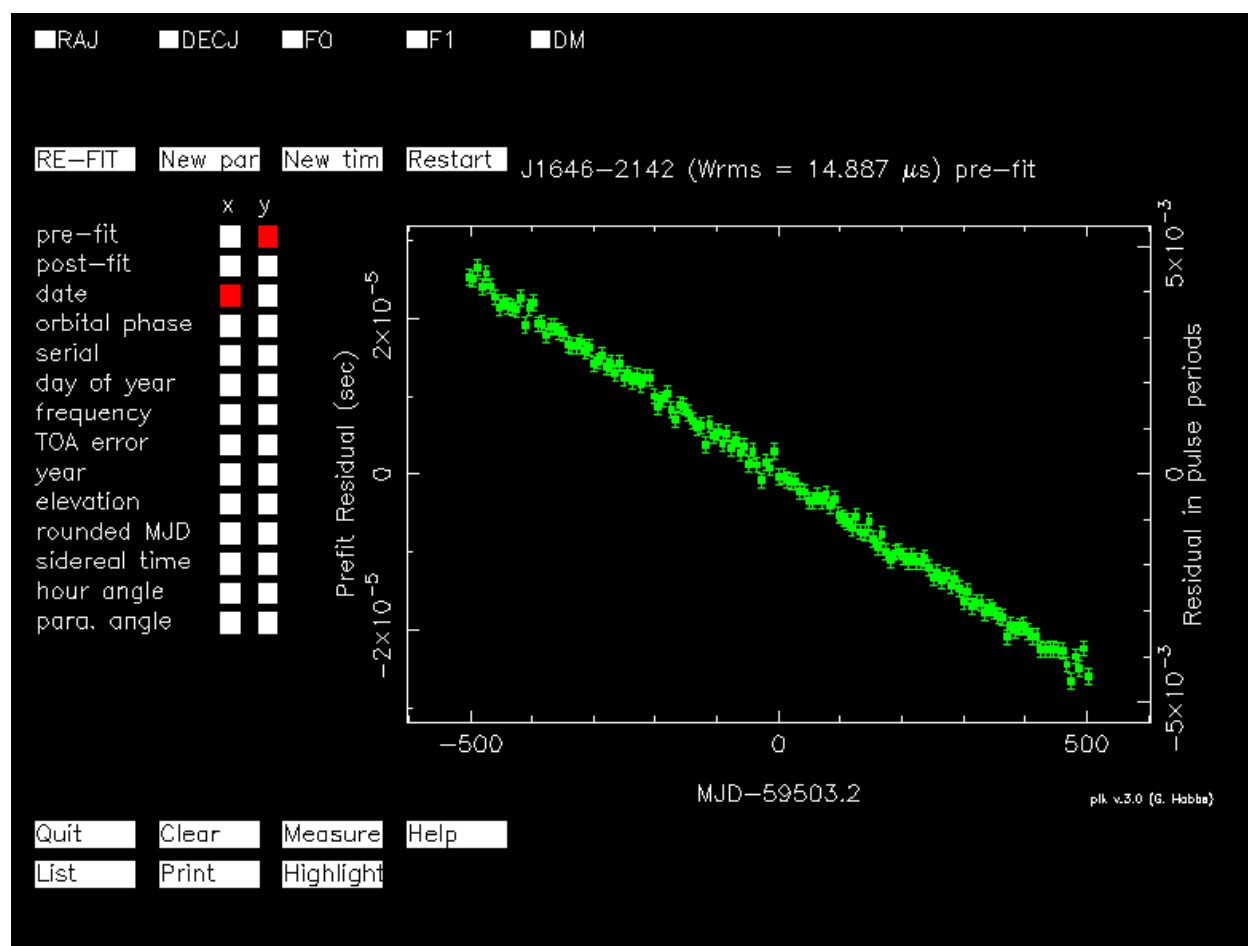
Again, if you fit for the F0, you will get the randomly sampled residuals around the zero line in the post-fit residuals.

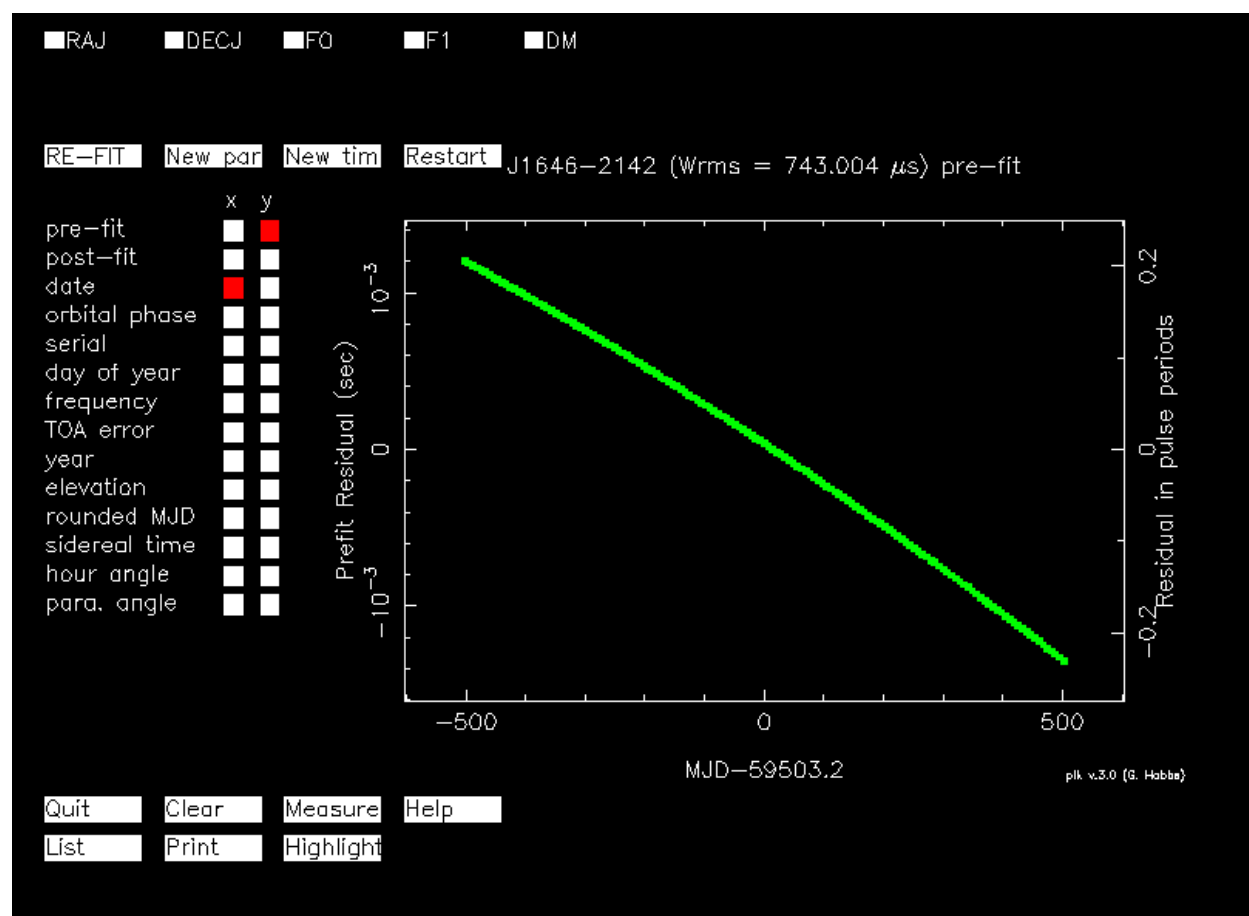
Try the same thing again with error in F1, and this time you'll notice a parabolic long-term variation like the one in the image below.

Understanding the pattern in the residuals caused by parameter error will assist us in solving a newly discovered pulsar. Once accurate (phase-coherent) solutions for a pulsar are obtained, one can establish its clock-like behaviour and predict its phases at any point in time.









5.3.4 Time to solve a Newly Discovered Pulsar

This time we will start with the real GMRT data set for a newly discovered pulsar J1244-4708 which is discovered in the GMRT High Resolution Southern Sky Survey.

This pulsar was localised through imaging of a single observational epoch which provides us with its RA and DEC which are accurate to a few arc-seconds. The F0 and DM values of the pulsar were estimated using data from an epoch observation in which the pulsar was strongly detected. We used the reciprocal of the period value (in seconds) given in the waterfall plot of that observation to calculate F0. Similarly, from the same plot, we noted the DM and epoch of the observation. Using these values, we created a parameter file for this pulsar (in this case, J1244-4708.par) that contains an initial guess of its parameters.

```
cat J1244-4708.par
```

```
RAJ    12:44:27.213 1  0.020
DECJ   -47.08.00.715 1  0.708
DM  74.870 1  0.0
PEPOCH 58559.797999747789
POSEPOCH 51544.0000000000
F0  0.70853408761 1  0.0
F1  0.0 1  0.0
```

In this exercise, we will use GMRT's incoherent array (IA) beam data. This pulsar has been observed 13 times over a baseline of 9 months. The GMRT IA beam data is first converted to "filterbank" format, then dedispersed and folded to "pfd" format. We'll start with the "pfd" files right away.

5.3.5 Time of arrival estimation

The "pfd.ps" files contain waterfall plots for all "pfd" files. The first step is to go through all of the profiles and find the cleanest one with the sufficiently bright integrated pulse. We will use that profile as our reference or template profile, assuming it is the true profile of the pulsar. Use the following command to view all the profiles.

```
evince *ps
```

After selecting the template profile, the next step is to estimate how many ToAs should be extracted from each observation. We will derive a single ToA for full observation for weak detections, two ToAs for mildly detected profiles, and four ToAs for bright detections. This approach will be adequate to find the solution in this specific case. To determine the number of ToAs for individual observations, check all the profiles again using the command above.

Let's say you've selected k ToAs for a particular observation A.pfd and B.pfd is your chosen epoch for the template. Then run the following command to get the ToAs for observation A.pfd.

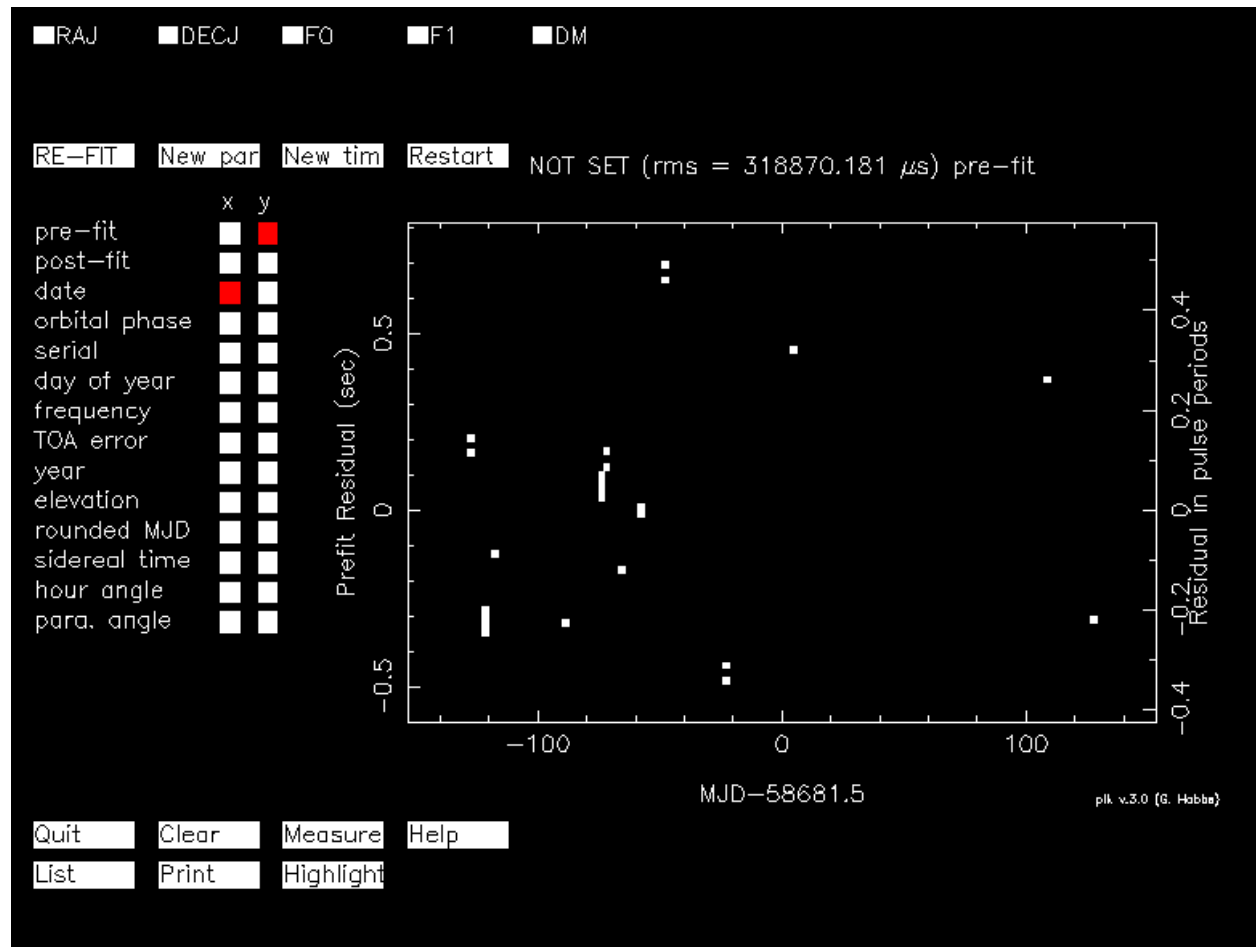
```
get_TOAs.py -n k -t B.bestprof A.pfd >> J1244-4708.tim
```

This command will find the ToAs for observation A.pfd by cross-correlating it with template B.bestprof. The ToAs will be saved in the J1244-4708.tim timing file.

The value of k varies depending on the detection strength of the pulsar for each observation. So, for each observation, use the same command, but change the file names A.pfd and the corresponding k value while keeping the template same (i.e., B.bestprof). Once all observations' ToAs have been generated, run the command below.

```
tempo2 -nofit -f J1244-4708.par J1244-4708.tim -gr plk
```

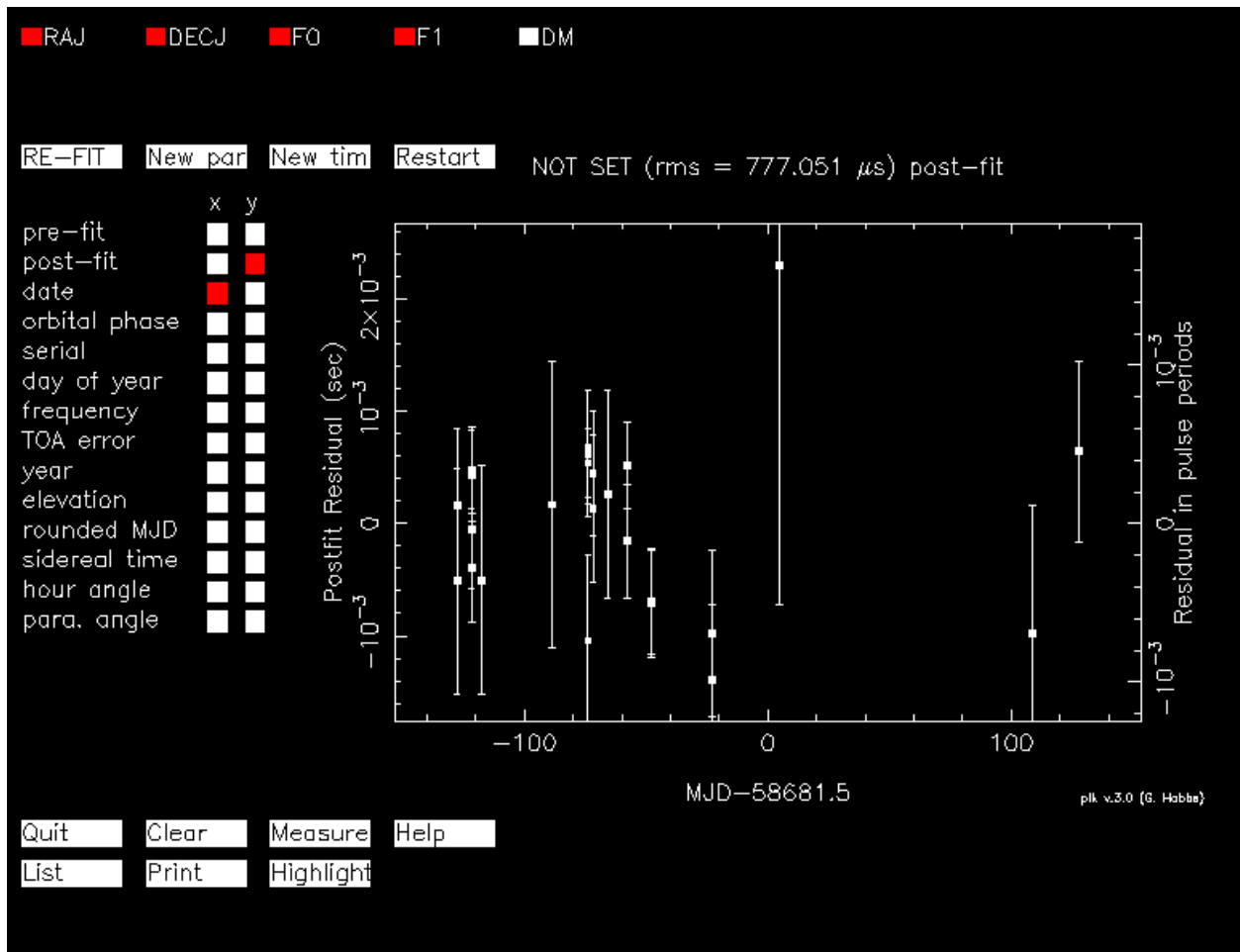
This command displays the difference (or residuals) between the observed ToAs (in the J1244-4708.tim file) and the predicted ToAs (predicted using J1244-4708.par file). Because the ToAs for a newly discovered pulsar are not phase-connected, we will not see any general systematic pattern in the residuals. This is illustrated in the figure below.



Now, take a sample of densely sampled points and start fitting from F0. Once you've identified a pattern in the ToAs, try fitting RA and DEC. Finally, you can try fitting F1, but keep in mind that the value of F1 for 9 months of data will be highly unreliable. Once phase coherent solutions are obtained, ToAs will exhibit near-random behaviour around the zero line, as shown in the figure below.

Examine the F0, F1, RA, and DEC values. Since RA and DEC are obtained through imaging, the post-fit RA and DEC should be within a few arc-seconds of the initial guess (i.e., pre-fit values). The F1 value after fitting should be negative and small (compared to error in F0). Below are the fitted values for this exercise.

Finally, on the graphical interface, select "new par" to create the new parameter file. Now that the pulsar's phase-coherent solution has been found, you will be able to predict the ToAs in future observations. The prediction accuracy improves as the number of observations and the timing baseline increase.



PARAMETER	Pre-fit	Post-fit	Uncertainty	Difference	Fit
RAJ (rad)	3.33565005922502	3.33558185282484	1.477e-05	-6.8206e-05	Y
RAJ (hms)	12:44:28.4808783	12:44:27.5429729	0.2031	-0.93791	
DECJ (rad)	-0.822628760812742	-0.822632550557029	3.5375e-06	-3.7897e-06	Y
DECJ (dms)	-47:07:59.36196	-47:08:00.14365	0.72966	-0.78169	
F0 (s^{-1})	0.708499626089485	0.708499629762667	7.9363e-10	3.6732e-09	Y
F1 (s^{-2})	0	-3.83029946268612e-16	8.2381e-17	-3.8303e-16	Y
PEPOCH (MJD)	58559.7979997478	58559.7979997478	0	0	N
POSEPOCH (MJD)	51544	51544	0	0	N
DMEPOCH (MJD)	58559.7979997478	58559.7979997478	0	0	N
DM (cm^{-3} pc)	74.87	74.87	0	0	N
START (MJD)	58552.8792882802	58553.8738780305	0	0.99459	N
FINISH (MJD)	58810.1202474477	58809.1241934994	0	-0.99605	N
TZRMJD	58686.4159663757	58686.4159663729	0	-2.8376e-09	N
TZRFRQ (MHz)	400	400	0	0	N
TZRSITE	r				
TRES	1135.9406112789	777.050813689493	0	-358.89	N
EPHVER	TEMP02	TEMP02	0	0	N

5.4 Acknowledgements and Credits

The **Searching for new pulsars** and **Timing of pulsars** sections of the tutorial were entirely contributed by Shubham Singh and Shyam Sunder, respectively. The data sets used in these parts were kindly provided by Dr. Jayanta Roy and Dr. Bhaswati Bhattacharyya. Yogesh Maan is responsible for the overall pulsar tutorial coordination as well as for making available the data used in the **Introduction** section.

CAPTURE PIPELINE