Giant Metre-wave Radio Telescope (GMRT)

NCRA-TIFR

The GMRT Beam Monitor (GBMon ver1.0)

Garvit Agarwal, Yashwant Gupta, Nilesh Raskar, Santaji Katore

July 5, 2022

Contents

1 Introduction

The GMRT Beam Monitor (GBMon) is a real-time monitoring tool for data obtained from the beamformer at the GMRT. Using this tool, its possible to monitor the beam data (for pulsars, RFI etc) simultaneously for 4 independent beams/frequencies. The workhorse behind the GBMon is the GMRT Pulsar Tool (GPTool) which was an already existing program at GMRT, capable of performing RFI filtering, dedispersion and folding of beamformer data in real-time. The GBMon uses a wrapper around the GPTool in order to use it in an automated way; to make decisions based on the current state of the observatory without any manual intervention.

2 Features

During its run, GBMon continuously checks the current state of the observation at GMRT; It checks (i) whether an observation has been started (more precisely, it checks whether a shared memory has been created for the beamformer data to be read), and (ii) given there is ongoing observation, whether the data acquisition system (DAS) has been started or not. If either of these conditions are not met, GBMon stops the previously running GPTool process (if any) and stays in the waiting mode (displaying the current time and waiting time) until the situation changes.

2.1 Dependencies

The GBMon wrapper itself is located at `/common-h10/gpuuser/pulsar/packages/gbmon_tests/gbmon.py` GBMon uses ver4.3.6_gbmon version of GPTool, located at `/common-h10/gpuuser/pulsar/packages/gptool_ver4.3.6_gbmon`. It sets the correct environment variables for GPTool by running `source /common-h10/gpuuser/pulsar/packages/scripts/presto_old_yg.bash`.

GPTool itself has the following dependencies: `pgplot (with libx11), openmp, psrcat, tempo/2` . In order to compile GPTool, use the command `make` in the 4.3.6_gbmon directory.

GBMon uses a python wrapper around GPTool and all the testing has been done for python ver2.7 . The python wrapper requires the following libraries- `subprocess, shm (source), re, signal, struct, __future__, time, datetime, os, sys`. GP-

Tool's plotgptoolsummary.py script requires the matplotlib library.

2.2 Input Parameters

Every observation/test at GMRT is identified by a unique code called 'GTAC code/project code' which is assigned by the observatory systems at the very beginning. If there are different subarrays at any instant, they are also assigned different GTAC codes. Using this code its possible to obtain the observation parameters of the subarray.

If GBMon finds an ongoing observation and data acquisition in start mode, it needs the GTAC code for the current beam. If GBMon is run through the GUI, the GTAC code is obtained from the TGC database. If it is run from the terminal, user can provide it as an command-line argument while running gbmon.py. If no argument is provided the wrapper acquires the GTAC code from the following command-

```
ssh gwbh6 /home/gpuuser/GWB/release/bin/get_gtac_code
```

GBMon uses this code to acquire the observation parameters via the following command:
`/home/gpuuser/GWB/release/bin/read_shm_hdr1 -c <gtac_code>`. This command reads the header of the shared memory where the real-time data will be read from during the observation.

When GPTool is run, it reads from a parameter file called `gptool.in`. This is where GPTool gets all the observation parameters, in addition to various options for RFI filtering, folding, plotting customization etc. 1) If GBMon is run via the GUI, the GUI creates a sub directory in `/common-h10/gpuuser/gbmon/` with timestamp, the GTAC code and beam number, and saves a gptool.in file in it. These parameters are used when GPTool is run. 2) If terminal, then the GBMon uses one of the gptool.in_beam<number> files, (depending on which beam GBMon is being run), to get the GPTool parameters. Hence the user can edit the gptool.in_beam<number> file, to change various customization parameters. If for some reason the above files cant be read, GBMon will use the `gptool.in_master` file which stays at a fixed location,
`/common-h10/gpuuser/pulsar/packages/gbmon_tests/`

Once the gptool.in file to be used is decided, the observation parameters obtained from the shared memory header are copied to the gptool.in file. At this point, GBmon checks whether the source name (as acquired from the shm header) is a standard

pulsar name, via regular expression pattern matching. If its not, and if the user has put '-1' for the period and DM in gptool.in, then default values of 100 ms and 0 are chosen for period and DM respectively. In addition, based on the bandwidth and the number of channels, 'the number of channels to flag at the band beginning and end' in gptool.in is set to $0.03 * no.of channels$, and the 'smoothing window' for narrow-band RFI mitigation is set to be 10 MHz wide.

Following this, GPTool is finally launched as a background process. Throughout the time GPTool is running, the wrapper checks for the observation status as described earlier and this loop-like behaviour continues until gbmon itself is closed.

2.3 GBMon Runtime Output

The primarily useful runtime output of the GBMon are the plots generated by the GPTool in real time (figure 1). In addition, GPTool also displays the important parameters related to the current run like source name, period, start and run time etc. The mode of display of these plots is decided by the gptool.in parameter 'pgplot display device type'. If its chosen to be '/png' then the individual plot frames will saved as png files. These plots can be seen live either (i) via a web interface (remotely), or (ii) on the host machines using a bash script (both are described below). However if instead of '/png' its '/xs', then GPTool displays the plots on the screen directly and the plots are not saved as images.

On the terminal, GBMon prints various statements indicating the steps completed and current status. Once GPTool is launched by GBMon, all information and parameters about its run is printed. Thereafter, GPTool shows the data block number that is being written to the shared memory by the beamformer and the block that is being read by GPTool.

2.4 Product Files

At the end of each of its run, GPTool itself generates several useful files, for flagging information, folded profiles etc. Some of these include the bandshape, folded profiles (filtered and unfiltered) etc. The plotgptoolsummary.py script of GPTool (automatically) generates a file named `summary.gpt.pdf` which contains information about the RFI quality in the data in the form of plots. All these output files can be found in a subdirectory in `/common-h10/gpuuser/gbmon/` (in the host machines), where the subdirectory is named in the following format: `<date>_<start`

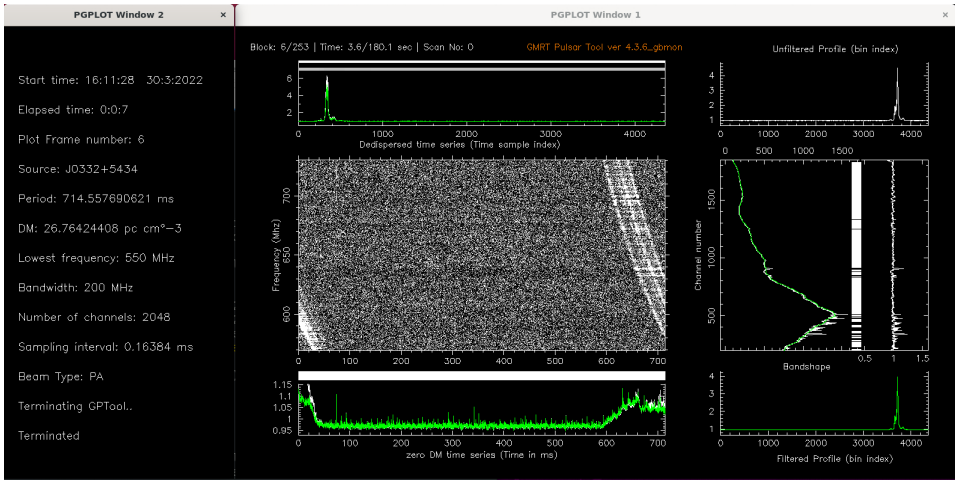


Figure 1 - The Plot window of GPTool

```
Looking for ongoing observation

Success. Found ongoing observation


GTAC code not provided
Getting GTAC code/Observation code

ssh gwbh6 /home/gpuuser/GWB/release/bin/get_gtac_code

GTAC code/Observation code being used: TST2543

Data aquisition is in start mode

Getting observation parameters and preparing gptool.in

Source name is not a standard Pulsar name.
Using Period: 100ms and DM: 0
Starting gptool in verbose mode..
```

Fig 2 - Part of GBMon's output on the terminal

`time>_<end time>_<gtac code>_<beam number>`. In this subdirectory, GBMon also saves the gptool.in file used in each GPTool run. If GBMon is run in the 'png mode', then the images are saved in the folder 'plots' in each subdirectory.

In addition, in a complete run of GBMon itself (having possibly multiple runs of GPTool), a log file is generated having a record of all of GBMon's standard output (including that of GPTool), and more information about the commands used inside the python wrapper. The log file name is of the following format- `<gbmon.log>_<date>_<start time>_<end time>` (if GBMon is run through the GUI, the logfile's name might not have the 'end time'.

2.5 Interactive Options (Command-line)

When GBMon is run directly from the terminal, the user gets an option to choose either 'quiet mode' or 'verbose mode'. The verbose mode prints all the default standard output of the wrapper and GPTool while in the quiet mode, all output of GPTool is suppressed.

At any instant, the user can interrupt GBMon run by pressing `Ctrl + C`. There are 2 possible scenarios, depending on whether the keyboard interrupt was given while (i) GPTool was running and (ii) while GPTool was not running and GBMon was simply in the waiting mode. In the first scenario, GPTool is first stopped and its product files are prepared and stored in the appropriate location as described above. Then the user has 2 options. Entering 0 will restart GPTool before which they can edit the gptool.in_beam<number> file if GBMon is run from the terminal in control room. Or inputting 1 will exit GBMon itself. In the second scenario, user can input 0 to go back to the waiting mode and 1 to exit GBMon.

2.6 GUI

GBMon can be fully run from a GUI as well. Detailed steps on how to use the GUI are provided in the SOP section below. The GUI reads the gptool.in_master file (at `/common-h10/gpuuser/pulsar/ packages/gbmon_tests/`) as the default file. The parameters provided by the user are then incorporated into a new gptool.in file in a subdirectory in `/common-h10/gpuuser/gbmon/` and then the gbmon wrapper is run. This is the same subdirectory that will contain the product files of GPTool.

2.7 Web UI

As described earlier, a remote user can use the web UI to monitor the entire live run of GPTool (as well as the past runs) by playing the images of the plots. The steps to follow to use the web UI are provided in the SOP section below. However to see a given run of GPTool on the web interface, before running first make sure to enter '/png' for the parameter 'pgplot display device type' in gptool.in whether its via the GUI or in the file gptool.in_beam<number>. Therefore, all the GPTool runs which have '/xs' as display device, will not be viewable by the web interface.

It is not possible to control GBMon from the web UI in its current state.

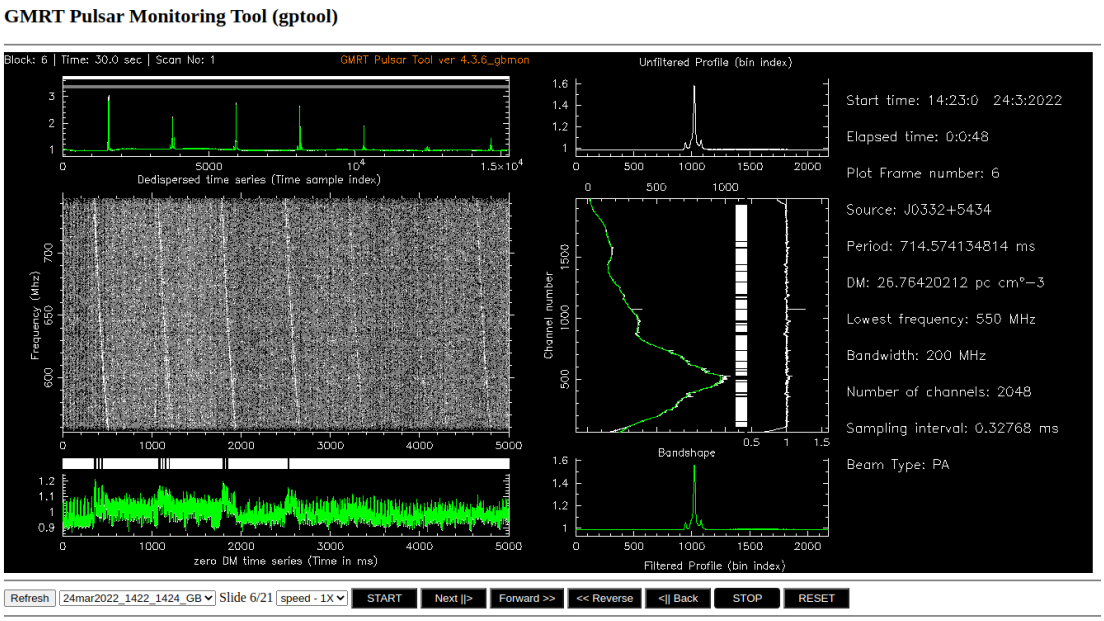


Figure 3 - The Web Interface

2.8 Alternative to web interface to see live png images

If GBMon is run with the '/png' parameter, its possible to view png images without the web interface, using a bash script located at `/common-h10/gpuuser/pulsar/packages/gbmon_tests/gbmon_png_viewer`. The steps to use it are given in the SOP section below. When these steps are followed, it will show the latest png image of GBMon as they are being created live. Hence, if there is no GBMon running in the png mode in the given beam, this script will just show the last png image that was created. Therefore, this cant show the user previous GBMon runs like the web interface. Also, the user cant go forward/backward in the png images.

2.9 Multiple Simultaneous Beams

Another feature which enhances GBMon's applicability is the ability of the user to monitor the GMRT's 4 sub-arrays/beams simultaneously. Its possible to have totally different configurations in different beams and GBMon will show independent results for each of them. All of the features described above have been implemented to work independently for each beam. Therefore, its possible to, say, run GBMon via the GUI in beam 1 in '/xs' mode, GUI with '/png' mode for beam 2 and use only the terminal in '/xs' mode for beam 3 for different sources and observational configurations.

To use GBMon for 'n' beams simultaneously, it needs to be launched n times separately. For the same reason, the steps mentioned in the standard operating procedure below allow the user to decide which beam they wish to monitor.

2.10 Changes made in GPTool

GBMon uses GPTool ver4.3.6_gbmon, which is a modified version of ver4.3.6. The following changes were made to this version for the purposes of GBMon:

1. In place of 'cout' in all the console output statements, a 'custom cout' is used. This custom cout not only prints statements to console/terminal, but it also writes the same statements to a file. The path to this file is provided by using the '-l <file path>' flag while running GPTool. GBMon gives the path to its log file using this flag to obtain GPTool's output.
2. Earlier in the real-time (SHM) mode of GPTool, several lines about status of

data blocks being read and written were printed for every data block. In the ver4.3.6_gbmon the verbosity has been reduced greatly, printing only the minimum required information and rewriting it on the same line.

3. GPTool originally had only the main pgplot window with plots of time series, folded profiles, bandshapes etc. For GBMon, another plot window called 'info plot' has been added, which displays the basic parameters of the source like name, period, DM etc and the current observation like number of channels, sampling interval, current and elapsed time etc. In addition, the main plot also displays the current scan number which shows the number of times the current source source has been observed in the current observation session. To provide GPTool with the scan number, the flag '-c <scan number>' is used while running it.

4. 2 extra parameters have been added to gptool.in.
   (i) 'width of the main pgplot window (in inches)': This can be used to decide the size of the main plot window. The height/width ratio is maintained at 0.618, so only the width needs to be changed.
   (ii) 'pgplot display device type': as described earlier, this determines whether the individual plot windows are saved as png files (which may be viewed using the web interface) or if they are only displayed in real-time. For the former mode, '/png' is to be used as the value of this parameter, and for the latter mode, '/xs' is to be used. If nothing is entered and or an invalid string is entered in gptool.in/GUI, then the default mode is the '/xs' mode.

3 Standard Operating Procedure

GBMon can be used to monitor multiple beams simultaneously. For each beam, GBMon needs to be launched separately. Hence the following steps need to be repeated for each beam.

3.1 GBMon via terminal

Follow the given steps to run GBMon via terminal:

1. Log into one of the beam host machines, GWBH7, GWBH8, GWBH9 or GWBH10.

2. Edit the file `gptool.in_beam<no.>` in `/common-h10/gpuuser/gbmon/` where `beam<no.>` corresponds to the current host machine. The following parameters

in this file need not be changed since they are read from the GMRT systems or the ATNF pulsar catalog: `beam mode, polarization mode, frequency band, bandwidth, sideband, number of channels, sampling interval, pulsar period and DM`. Rest of the parameters in gptool.in can be changed as per the need.

3. Run the command: `python /common-h10/gpuuser/pulsar/packages/gbmon_tests/gbmon.py`. This will launch GBMon in the beam corresponding to the current host machine.

Step 3 can be done from any directory in any of the beam host machines. At any time during GBMon's run, user can interrupt using `Ctrl + C`. At the end of each GPTool run after pressing `Ctrl+C` (GBMon may keep running) the output and log files generated by GBMon will be saved in a suitable directory in `/common-h10/gpuuser/gbmon/`, identified by the date, timestamps, observation/gtac code and the beam number in that order eg 22feb2022_1834_1835_TST2523_beam1.

3.2 GBMon via the GUI

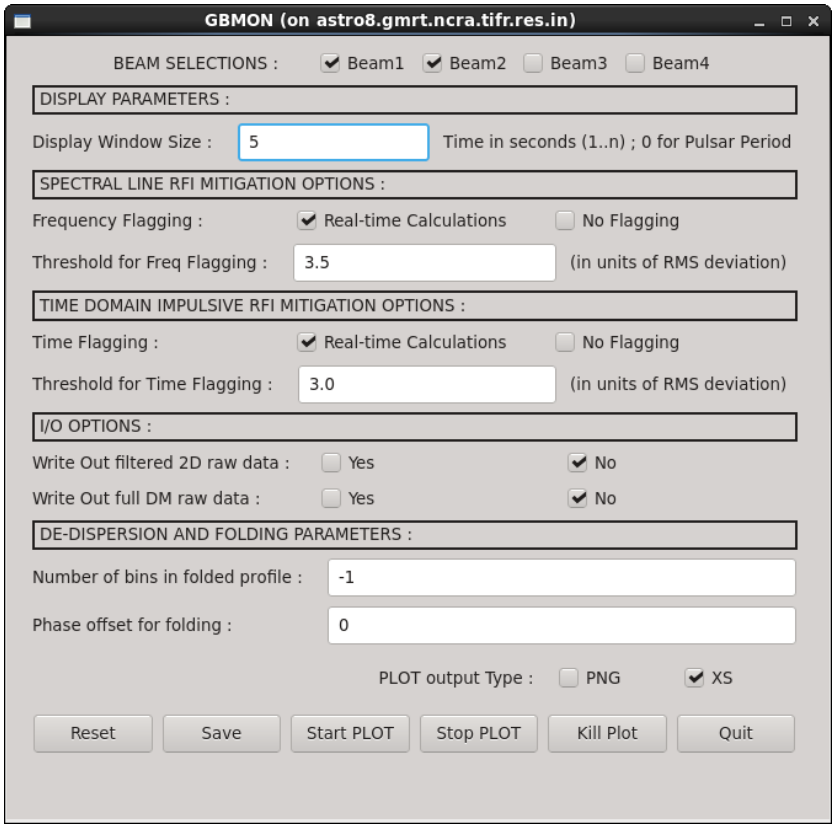
The GUIs for each beam are independent, and these can be run simultaneously without affecting each other. Follow these steps:

1. Log into the astro8 machine [ssh -X observer@astro8].
2. Depending on the desired beam, enter the command `gbmon_gui_beam<no.>`. This will open the GUI window, which looks like the following-
3. At the top 'beam selections' shows the beams that are currently acquiring data. This will be preset as per real-time beam parameters selections from TGC. This is just for information and it DOES NOT indicate the beams on which GBMon is running. That is determined by the beam no. used in launching the GUI.

   TGC commands to link beam selection to GUI:

   ```
   set_gwb_beam(beam1='PA',beam2='IA',beam3='None',beam4='None')
   associate_gwb_beam(1, [1, 2])
   ```

4. Here is a description of the parameters that the user can set-
   - Display window size: the duration of time considered for each plot window. This value is preset to 5 seconds. Enter 0 to set this parameter to the pulsar period.



The GUI

- Frequency Flagging: Real-time narrow band flagging ON or OFF (No flagging).
- Threshold for frequency flagging: value in units of RMS deviation.
- Time Flagging: Real-time wide band flagging ON or OFF (No flagging).
- Threshold for time flagging: value in units of RMS deviation.
- Write Out filtered 2D raw data: Yes or No.
- Write Out Full-DM raw data: Yes or No.
- Number of bins in folded profile (-1 for native resolution).
- Phase Offset for folding.
- Plot Output Type: 'PNG' saves the GPTool plots as images which can be viewed on the web interface (see below). 'XS' generates plots on the current desktop terminal/workspace. Only one option can be selected at a time.

5. Here is a description of the 6 buttons at the bottom of the GUI-

- Reset - Resets the changed parameters to default values.
- Save - Saves the parameters selection on GUI.
- Start Plot - Start GBMon for the beam mentioned in the GUI launch command (and not for all the beams selected in the 'beam selections'). It will open a xterminal window which will print GBMon run status.
- Stop Plot - currently disabled.

- Kill Plot - Kills the GBMon run of the current beam (abruptly).
- Quit - quit the interface.

As of now, since the 'Stop Plot' button is disabled, the user can stop GBMon by clicking on the xterminal window of the desired beam (opened after clicking on Start Plot) and pressing Ctrl+C. After this, click on 'Kill Plot'. This brings the GUI back to the original state, ready to be used again. Please note that directly clicking on Kill Plot button will apruptly stop GBMon and the saved directory and some files will not be saved correctly.

3.3 Using the web interface

Once GBMon is run from either the terminal or from the GUI (in any beam), the user can remotely monitor GBMon's live output plots, as well as replay past GBMon runs using the web interface. To see a given run of GPTool on the web interface, follow the given steps:

1. Open the webpage: https://www.gmrt.ncra.tifr.res.in/∼astrosupp/gbmon/psr.html
2. Click on 'select profile'.
3. (i) To view older runs of GBMon, select the item in the list with the appropriate timestamps, gtac code and beam number. Then click on 'Start' button to play the GBMon plots from the start.
   (ii) To view the current live run of GBMon, click on 'refresh', then select '`realtime_beam<number>`' for a particular beam. Then click on 'Live' button to view the real-time display of GBMon.
   Other buttons on the webpage can be used to navigate between the individual plot frames as desired. Note: The web interface shows only those GPTool runs for which the gptool.in parameter 'pgplot display device type' is set to '/png'. This can be done either by editing gptool.in_beam<no.> (if GBMon is being run via terminal) or by using the GUI.

   Tip: If the webpage is not showing updated profile lists to the current date and times for each running beam then please ensure that **lsyncd** is running on gwbh10 machine.

   ```
   lsyncd -delay 1 -rsync /common-h10/gpuuser/gbmon/ gpuuser@gwbh6:
   /gwbifrdata/SUMMARY/gbmon/
   ```

3.4 Another way to see png images without the web interface

If GBMon is run with the '/png' parameter then the live images can be seen through the terminal without the web interface. Knowing the beam number, follow these steps:

1. Log in to one of the beam host machines, GWBH7, GWBH8, GWBH9 or GWBH10.
2. Enter the following command: `python /common-h10/gpuuser/pulsar/packages/gbmon_tests/gbmon_png_viewer /common-h10/gpuuser/gbmon/realtime_beam<number>`

This will show the live png images being created. User cant view saved png images of previous GBMon runs through this.

3.5 Scope for control room user and remote user

Before GBMon can be run in any way, the user (through the telescope operators) needs to set up data acquisition for the desired source with the correct parameters. For a user in the control room, GBMon can be run in the full capacity via either terminal or the GUI as explained above. In the 'xs' mode, the plots will be displayed on the current desktop while in the 'png' mode, the user can open the web interface to see the plots. The user also has full control over the flow of GBMon run.

For a remote user however, its only possible to see the GBMon plots via the web interface. The user cant launch GBMon or control it. Hence GBMon has to be run with the 'png' option by the operators, and then the remote user can follow the steps mentioned in 'GBMon via the web interface'.

4 Limitations and Open Problems

The following are some current limitations of GBMon, in addition to some further possible applications.

1. Currently its not possible to control gbmon from the web interface, like start or stop the run or input gptool.in parameters.

2. Only standard names of pulsars are recognized by GBMon. Hence if a non-standard name of a pulsar is fed in the observatory systems, GBMon would use value of 100 ms for the period and 0 for the DM.

3. A possible application is obtaining rfi characteristics of individual antennas, by running GBMon on individual antennas in an automated way using all the 4 beams.

4. The GUI doesnt have a working 'Stop Plot' button. It is supposed to do exactly what Ctrl+C does in the terminal mode. Right now, user has to select the xterminal window and press Ctrl+C in the GUI mode.