CYGNSS DDM Processor Basics, V2
Scott Gleason
September 2017

1) Summary:

This document provides some very basic information on building and running the CYGNSS DDM Processor. It is not meant to be a user manual, and it is assumed that people building, running and modifying this code have a good knowledge of C/C++ programing and Matlab/Octave.

The code is released under a GPLv3 open source license for the benefit of the CYGNSS Science Team. Please provide feedback on bug fixes, improvements and general comments and feel free to modify the code as you wish.

The CYGNSS DDM Processor generates full DDMs over the full duration of CYGNSS raw IF data collections. It uses a “delay” domain FFT technique to perform correlations at all delay samples using a frequency domain multiplication. The delay sampling is configurable based on decimation of the sampling rate (down to 1/16 chip spacing) and the Doppler processing range and non-Coherent integration duration is completely configurable. The program will process over the time duration of the entire collection (as configured) or until it runs out of data.

2) Build Information:

The Beta release of the CYGNSS DDM Processor was designed and built on a Linux Ubuntu laptop using the free (non-commercial) version of Qt Creator (Version 3.6.0). On Linux Qt uses the GCC compiler. Different versions of Qt can be downloaded from https://www.qt.io/download/. However, most of the code (other than the GUI interface) is written in straight C and could be ported to another environment and/or run without a GUI interface with some work.

The program requires the fftw3 development FFT library (libfftw3-dev). In Linux, this can be downloaded and installed using most Linux package managers (i.e. synaptic on Ubuntu).

3) Program File Summary:

1) CYGNSS_DDM_Processor.pro
This is Qt's version of a project file, which resembles a Makefile and is used to build the project.
2) main.cpp
Created by Qt to generate the mainwindow Class.
3) mainwindow.h
Created by Qt as the header file for the mainwindow Class.
4) mainwindow.cpp
Reads in all information from the GUI, outputs messages to the GUI. Performs three high level loops: a) a full PRN loop if cold search option selected (otherwise only a single PRN loop is performed), b) the time loop through the raw IF data file based on GUI parameters and c) the 1ms non-coherent integration loop. Every 1ms a full single “look” DDM is generated and averaged in the routines below. The loops above were included in this file as it was easier to interface with the GUI in this function in real-time as the program run (i.e. display run-time messages).

[The following files are really C files, but named .cpp for easier integration with the C++ GUI]
4) CYGNSS_DDMP_Main.cpp
Performs various initialization, including reading and parsing the external configuration file and indexing into and reading the raw ID data file (and bit unpacking of the raw IF data).

5) CYGNSS_DDMP_Processing.cpp
Performs Doppler loop processing (calls fftw_acquire), non coherent averaging and logging of processed DDM to binary file. Note that “Cropped_DDM” and “IDW” functions not tested. All testing so far has been done logging “Full_DDMs”, for this reason the other two options are “greyed out” on the GUI.
6) fftw_acquire.cpp
Performs full delay based FFT correlation on 1ms sample vector using multiplication in the frequency domain with the FFT of the satellite PRN code. Heritage from the fastgps receiver. If logging of raw I and Q correlations is desired, this could be done at the end of the fft_acquire function.
7) code_table.cpp
Array of GPS L1 C/A PRN codes for all 32 satellites. Heritage from the fastgps receiver (http://www.gnssapplications.org/chapter5.html).

4) Basic Initialization

On the “Per DDM Processing” tab.
Output Type: “greyed out”. Is always “Full DDM” in that it outputs all delay bins from FFT.
Number of Channels Logged. Currently for the DMR on-orbit collections this is 3 channels. Only change this if collection configuration was specified differently in met dat a file. See Example document for processing met data.
Delay Bins to Crop. ”Greyed out”, currently not tested.
Load Configuration File. If you are not using the default configuration file, your own file can be loaded using this button.
Delay Spacing (Chips). This determines how the input samples are decimated to define delay bin spacings. These spacings are determined by the sampling frequency. The larger the spacing the faster the program will run (as the FFT vector is shorter).

On the “**Advanced Signal Search**” Tab, the following parameters are initialized
1) Sampling Frequency. The default is 16036200 Hz for the CYGNSS DMR (and initial data collected in the lab shows it actually is fairly close to that). This field exists so adjustments for the actual on-orbit sampling frequency can be made if necessary.
2) Intermediate Frequency. The center (0 Doppler) mixed down frequency of the sampled signal. The default for the CYGNSS DMR is 3872400 Hz.

Currently the program does not read in and parse the DMR meta data that links the raw IF samples to the GPS time they were collected (and hence allows linking the raw DDMs to the rest of the SOC L1 and L2 netCDF file metadata). However, there is an Octave/Matlab script that is explained in Processing_CYGNSS_RawIF_Collection_Example.doc That describes how to process the raw IF meta data file and link it to the Level 1 netcdf file information to initialize the DDM Processor.

3) GPS Week at Start of File. “Greyed out” not used.
4) GPS Week at Start of File.“Greyed out” not used.

5) Run Options

Configuration File

The default configuration file is “CYGNSS_DDM_config.dat” in the “Configuration_Files” directory. There are currently only two fields in the file, one required the other optional.

The required field specifies the raw IF data file name to process. The “F” field. This should

include the full path to the raw IF data to process. For example,
% FM8 Harvey data
^F Input_DMR_Data/cyg08_raw_if_20170825_141629_data.bin

The optional field is the PRN Doppler entry described below. The “D” field. This allows you to specify the PRN to process and the center Doppler over a series of consecutive seconds. Details on using this option are included in Processing_CYGNSS_RawIF_Collection_Example.doc

Only entries preceded by a “^” are processed and there should only be only one entry commented in for each field (if not it will use the last one it finds). All other lead line characters are ignored and are essentially comments. Do not use “^” characters anywhere else in the file.

Run Modes

1) Normal Single PRN Processing. In this option all GUI fields are applied during processing. Including on the “Per DDM Processing” tab, (PRN, Doppler range, non-coh interval, antenna, etc) and on the “Time Series Processing” tab (seconds in file to process). This is a good option for processing DDMs over a couple seconds, where the Doppler center of the signal remains reasonably constant.

2) Normal Single PRN File Doppler Processing. In this option all GUI fields are applied during processing except the “PRN” field and the Doppler fields (Start Doppler, End Doppler, Doppler Step). To run in this mode perform the following,

a) On the “Advanced Signal Search” tab, check only the “Use PRN/Dopplers in File” box.
b) Define an entry in the configuration file along the lines of,
^D 28 3 5000 200 1500 -27 0

The format of this line is,
“^” key character, without this the parser will skip this line.
“D” specifies a PRN/Doppler command sequence to use (if box on GUI is checked)
“28” PRN to process
“3” Process antenna 3 (i.e. nadir 2) [antenna 1 = Zenith, antenna 2 = Nadir 1, antenna 3 = Nadir 2]
“5000” Process -/+ 5000Hz around the following center Doppler frequencies
“200” Process in 200Hz Doppler steps
“1500” Center Doppler frequency offset at first processed second
“-27” Doppler rate of change per second. i.e. it will adjust center Doppler by this amount every second.
“0” Doppler second derivative … just in case.

3) Full 32 satellite PRN search. This can be useful to perform a thorough search of all PRNs in the data over a wide Doppler search range. Depending on the configuration of the search range (in particular the range and step of Dopplers and the number of non-coherent integrations) this search can take some time. For a 32 PRN Doppler search over -/+ 40kHz at 500Hz steps and 500ms of non-coherent integration takes about 15 minutes. Searching for direct signals on the Zenith antenna only a few non-coherent integrations is needed (1 to 10 usually) to detect signals.

a) On the “Advanced Signal Search” tab, check only the “All PRN Cold Search” box.
b) On the “Time Series Processing” tab, make sure that the Start and End times are the same (i.e. it only processes at one second). I normally make both start and end times 0, the start of the raw IF file data.

c) On the “Per DDM Processing” tab, configure the Doppler Parameters and the Non-coherent integrations as desired. For example, Start Doppler = -40000, End Doppler = 40000, Doppler Step = 500 and Non-Coherent 1ms Integrations = 500 usually detects most GSS signals. On-orbit, with weaker signals 1000 Non-Coherent Integrations may be better (but take twice as long). The Non-Coherent Integrations can be set higher than 1000, but with longer intervals some blurring of the signals may occur.

For this option, all GUI fields (except PRN) are applied as set on the GUI.

6) Post-Processing Output Files

When the CYGNSS DDM Processor is finished processing all DDMs that is was configured to do, it will output a binary file of all the processed DDMs in the execution directory called,

Processed_DDMs.bin

This file can be processed with an Octave script (which should work in Matlab but has not been tested). The Octave script to process the output DDMs is called,

plot_FFT_DDMs_binary1.m

And is included in the code release. This file performs peak detection and cropping of the Full DDMs and displays both the cropped DDM image and a series of delay waveforms. (and the Full DDM if you really want to comment that in!).

For more details on processing the Full DDM outputs see
Processing_CYGNSS_RawIF_Collection_Example.doc