Modified GB Mapmaker: Documentation

Suinne Lee

December 29, 2022

1 Introduction

This is a documentation on the TOAST mapmaker that has been modified for mapmaking simulations with GroundBird. Here we look at how each part of the code works, what kind of input data are necessary, and what current problems we face.

2 Detailed outline of the code

2.1 Data formulation

2.1.1 Import tools

We import the required tools here. They are:
1) todgb.py
2) TOAST package

2.1.2 Obtaining data

We have to give the following inputs to create a Data class for the simulations.

- Focal Plane(Credits: Seongsu Kim)

  Here we read the following 3 files (.txt tables):

  - Zp (pointing)

    The columns are: detector number, x, y, z components of the pointing vector
  - fp (position on the focal plane)

    The columns are: detector number, x, y, coordinates on focal plane

- angle (polarization angle)

  The columns are: detector number, angle on the focal plane, sky polarization angle

The code then computes detector polarization and logs it into the data.obs class. Here we must note the way the detector polarization is defined. For the GB focal plane layout, each detector is manually assigned its proper polarization. However, when sampling a subset of these detectors, a modification may be needed.

- Weather(Credits: Suinne Lee)

  Here we read from a weather file (.fits table) Weather data for Teide Observatory in the Canary Islands can be found here, and you can use the weather writing module to write your own weather data.

- Scan Parameters Scan-related input parameters include:

  - Sample rate (s^{-1})

    Take care not to have a large GCD for sample rate and scan rate. It is best if they are relatively prime. (Else there may be unobserved patches in the sky)

  - Scan duration (s)
  - Scan rate (rotation speed, degrees/s)

    GB rotates at 20 rpm (120 degrees/s). The original TOAST mapmaking tool uses an alternating CES scheme, and todgb is necessary to incorporate a full rotation scheme that GB uses.

  - Scan acceleration (degrees/s^2)

    In the GB rotation scheme, scan acceleration is set to 0.

Here the code consists of cells that define data writing functions and cells that execute them. Through this process we write these inputs into a data.obs class so the TOAST mapmaker can read them when simulating.

2.2 Mapmaking

Here we determine some parameters (mostly related to the resolution of the map):

- nside of the input map

- nside in the mapmaking process Cf) for the nsides, we used 64 or 128.
- nnz (we used 3)

Then, the code defines mapmaking arguments based on the scan parameters we have input in the last section. Here the telescope's gain of 3×10^{-5} has been taken from POLARBEAR.

2.2.1 Constructing input signal

In this simulation, we generate a simulated CMB signal (or an arbitrarily generated CLS signal) to use as the input signal. The input parameters are as follows:

- Hubble Constant (km/s/Mpc)
- Unit of the input signal (unit of temperature, here we used μK)
- L_{max} (Maximum correlation length, 1024 used)
- nside and FWHM (determines resolution and blur)

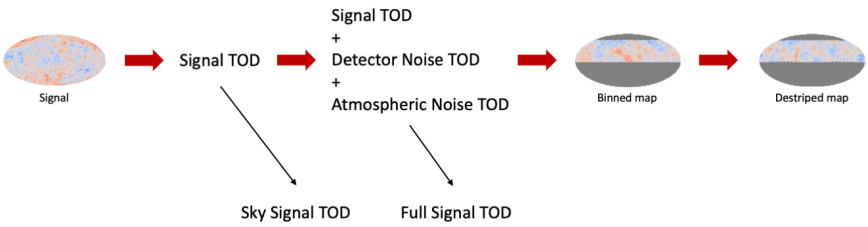
This is if you use CAMB to create a simulated CMB. Alternatively, you can use a pre-existing map by calling the map directly into the code in place of our created signal file. Another alternative is to use a Gaussian angular power spectrum of arbitrary scale.

2.2.2 Hitmap

Using data.obs, we write a hitmap computed from the detector pointing. The code checks if the number of hits equals the number of total samples and lets you know if there is a problem.

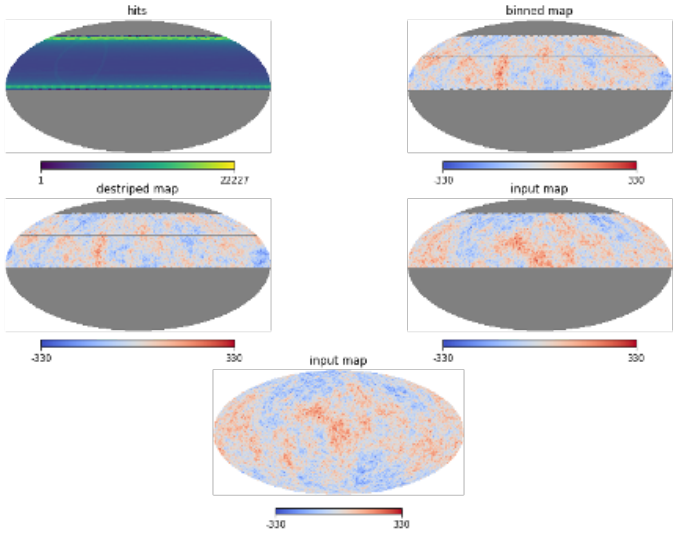
2.2.3 Scanning

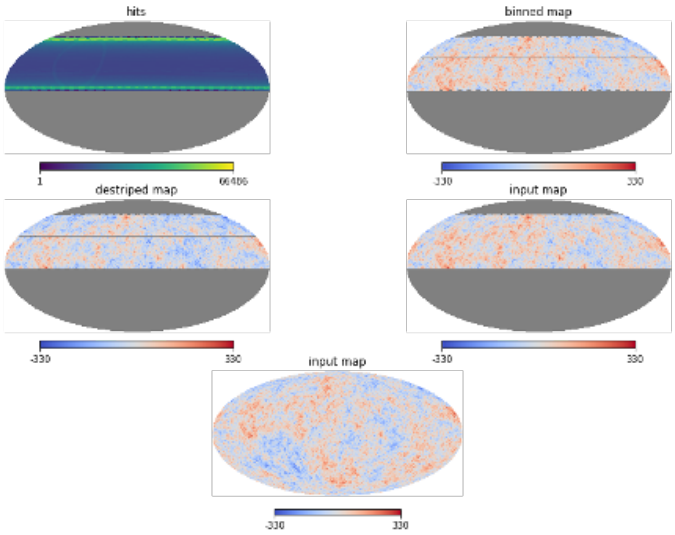
In this part our virtual GB scans the simulated input signal. The noise (detector noise and atmospheric noise) is also added, in the following scheme:



2.2.4 Run mapmaker

With such procedures, the mapmaker computes a binned map at the area of your hits, and in turn computes a destriped map. Simulating for more than one day is recommended so mapmaker has enough hits to compute the binned map.





The maps above were simulated for 1 day and 3 day scans respectively, which are both still too short compared to what's normally done to be able to effectively remove white noise. Here the relative scales of the signal and the noises (from the detector and atmosphere) were the same, with the noise being around 2 to 3 times greater than the signal.

2.3 Compact Version

I have also included a compact version of the mapmaking code that makes changing input factors very easy. Here all the functions are defined in the first half, and toward the last few blocks you will get to input factors including the scan related parameters and noise related parameters (namely the NET of the detector noise and the telescope's gain)

3 Noise Scale and Noise Removal

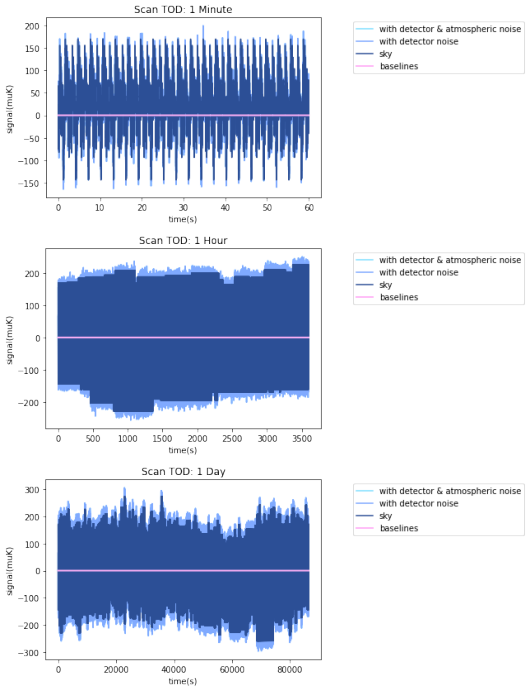
3.1 Noise Scale

Binning and destriping may yield maps different from the input map if the scale of the noise is much greater than the scale of the signal.

The telescope uses a gain of 3×10^{-5} (calibrated from POLARBEAR data) when converting the signal to a voltage. Naturally, this gain is common for the signal and atmospheric noise, and a change in this gain would only change the relative scale of the detector noise. However, since the code works separately when generating the signal and atmospheric noise, this gain only

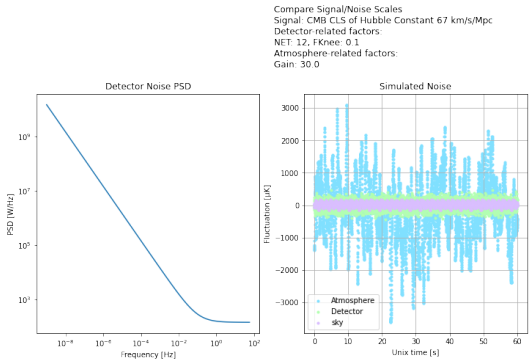
changes the scale of the atmospheric noise. The signal can be changed by altering the Hubble constant or manually adjusting the scale of the input CLS.

Since the atmospheric simulation gives back atmospheric fluctuation in Kelvins, I have also added a factor 10^6 to account for the change in unit to μK.



In the TOD as shown in the figure, the noise scale remains more or less the same as the timescale varies (unless the timescale of the scan is smaller than a single fluctuation) But in mapmaking, the more number of times you have acquired measurements for a given point in space, the smaller the white noise gets: as when you compute the point's value in the map, you end up averaging out the noise and getting a decrease by $1/\sqrt{n}$. Unfortunately, due to technical difficulties of running the mapmaker for a prolonged scan time, I could not include these results here.

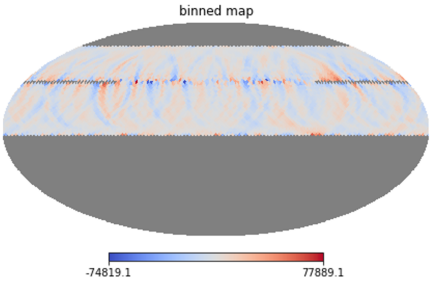
At the GB-appropriate setting, where the detector's $F_{knee} = 0.1$ and NET = 12, and the telescope has a gain of $3{\times}10^{-5}$, the scales of the signals and noise are as follows:



- Scale of the signal: 10^{-4} K (300 muK)
- Scale of detector noise: 10^{-4} K (About twice the signal)
- Scale of atmospheric noise: 10^{-3} K

3.2 Noise Removal - Destriping

When the detector noise is of a larger scale than the signal, ring shapes are visible because of the temporal correlation in the detector noise.

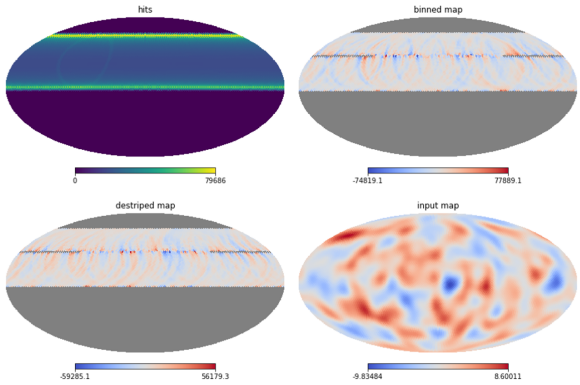


The figure shows a binned map with ring patterns due to detector noise. This was simulated with a very large scale atmospheric noise compared to the signal.

As mentioned earlier, scanning for a longer period (n times more samples) reduces the noise by $\sqrt{n}$, since n times more samples are used to compute the measurement at a certain point in space. White noise can be treated this way. The detector noise is time-dependent, and does not change largely with the change in rotation speed. The signal is a function of position in space, and atmospheric noise is both space and time dependent. A high speed scanning strategy therefore makes it easier to separate the slow drift (the detector's 1/f noise) and also suppresses the time dependence of

atmospheric noise. However, increasing the scanning speed alone cannot infinitely simplify the removal of atmospheric noise, and it is presumed the power spectral density and spatial correlation spectrum (Kolmogorov Spectrum) are used to separate atmospheric fluctuations from the signal.

With very large scale noises (and a relatively short scan period of about a day) destriping is not done very efficiently. In this figure, the atmospheric noise is very big compared to the signal (which was arbitrarily chosen to be around 10^{-6}.) The atmospheric noise here corresponds to a $1K$ fluctuation with realistic CMB signal.



Some Problems: However, unlike what I anticipated, it seemed that the TOAST mapmaker destripes atmospheric noise more effectively than detector noise of the same scale.

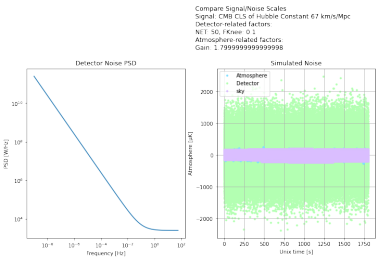


Figure 1: Mostly detector noise

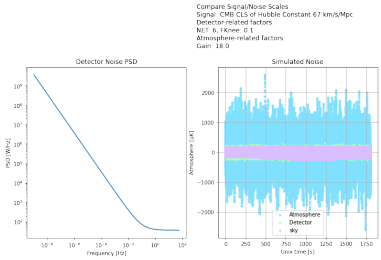


Figure 2: Mostly atmospheric noise

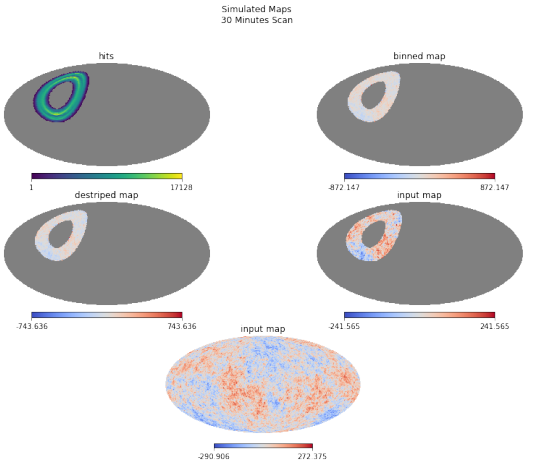


Figure 3: Detector noise destriped

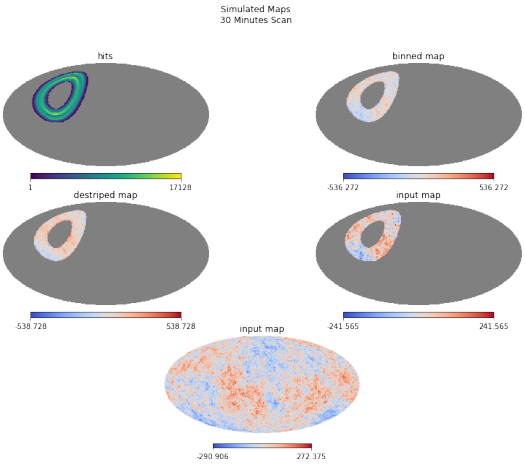
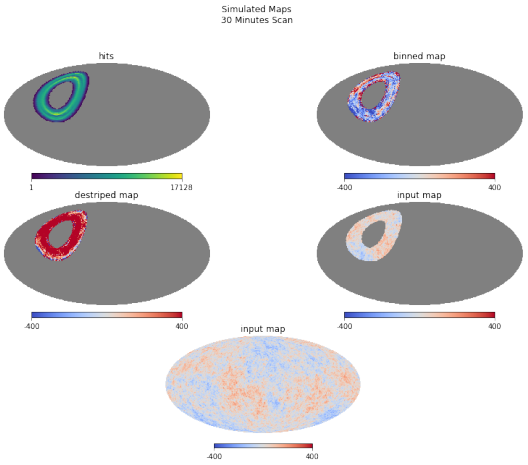


Figure 4: Atmospheric noise destriped

This may still be due to some extraordinarily large deviations that remain after destriping, some of which give the resulting destriped map an offset in the fluctuation (see figure below.) It would be much more significant to study these maps while running them for longer scan durations, which is hard to realize in my mapmaking environment. Perhaps this could be done using the MADAM mapmaker instead, which is much faster.



Credits: CMB TOAST tools are used in the mapmaking process. This code is a modified version of the mapmaking tutorial in the TOAST package.
The focal plane data was documented by Seongsu Kim.
The data class writing code was partly written and modified by Seongsu Kim to incorporate GB focal plane information.
The weather data and weather writing module was written by Suinne Lee, and uses sources from MERRA-2, Stella, Gaulli and on-site measurements at the IAC.
The code uses todgb.py to incorporate GB's scan scheme, written by Junya Suzuki.
The overall code was modified by Suinne Lee.