The LiteBIRD Simulation Framework & the LiteBIRD Instrument Model (IMO)

May, 22nd 2023

Maurizio Tomasi (Università degli studi di Milano)

The LiteBIRD Simulation Framework

Purpose of the framework

- Tool to simulate the three instruments (LFT, MFT, HFT)
- Developed by the Simulation Team (chair: Maurizio Tomasi, co-chair: Luca Pagano)
- Written in Python (~10,000 lines of code as of May 2023)
- Extensive documentation (~23,000 words)
- Full interface to the IMO database (see later)
- Used by the Simulation Production Team to run the end-to-end (E2E) simulations (see Giuseppe Puglisi's presentation)

The Simulation Team

- Established during the F2F meeting in Garching (December 2019)
- 39 people are currently registered to the mailing list
- Several of them are from Italy and regularly attend the telecons (minutes are available at https://wiki.kek.jp/display/cmb/LiteBIRD+simulation+team)
- Many PhD and post-docs: the average age of the team is young!

Which framework?

- We considered the following options:
  i. Planck LevelS was adapted for LiteBIRD by Davide Maino and used for some simulations, but it is written in C++ and many people prefer to work using Python, as it is easy to use it interactively. Moreover, LevelS contains many “Planck-isms”
  ii. TOAST (https://github.com/hpc4cmb/toast) is the de-facto standard for the simulation and data analysis experiments. It's written in Python 😀 and has many modules 😀😀, but it's very complex 🙁 and lacks documentation 🙁🙁
- We decided to develop a new framework while taking advantage of TOAST (which is a dependency of our framework and is being used internally for a few tasks)

What has been done so far

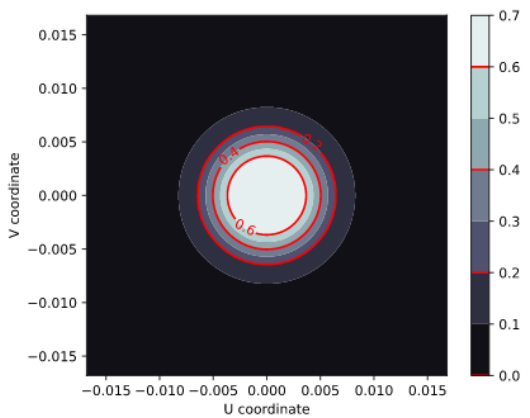
In 2020–2023, we have developed the following features:

1. Full documentation for **everything** (students love it!)
2. Powerful tool to produce automatic reports in HTML, with plots, tables, and equations
3. Simulation of the spacecraft's scanning strategy and orbit
4. Simulation of the observation of a realistic sky map via PySM3
5. Noise generation (white and 1/f noise)
6. Simulation of the signal of the CMB dipole, using different models
7. Simulation of an HWP modeled after a paper by S. Giardiello et al. (2022)

Parameter	Value
Pixels used in the fit	112
Integration time	15166.0 s

Error on beam estimation

This is a representation of the model of the main beam used in the simulation, in (u, v) coordinates, where $u = \sin\theta\,\cos\phi$ and $v = \sin\theta\,\sin\phi$:



Here is the result of the estimate of $\delta\gamma$, using the following formula:

$$\delta\gamma(\vec{r}) = \frac{\Omega_b \cdot \mathrm{WN}}{\pi r_{\mathrm{pl}}^2 T_{\mathrm{br,pl}} \sqrt{\tau}} \sqrt{\frac{1}{\sum_{i=1}^{N} \left(\frac{1}{4\pi d_{\mathrm{pl}}^2(t_i)}\right)^2}},$$

where N is the number of samples observed along direction $ `\vec r` $, WN is the white noise level expressed as $\mathrm{K} \cdot \sqrt{s}$, r_{pl} is the planet's radius, $d_{\mathrm{pl}}(t)$ is the planet-spacecraft distance at time t, and τ is the sample integration time (assumed equal for all the samples).

HEAD

- Mbs supports generic bandpasses and can generate solar dipole #227
- Improve the support for multiple TODs in the same `Observation` #225
- Remove bandpass-related warnings #236
- Improve the documentation #231
- Use Poetry instead of Pip to specify the dependencies for the documentation #237

Version 0.9.0

- Some memory optimization in pointing production #222, coordinate rotation and noise #223
- Implement new methods in the `Simulation` class: `fill_tods`, `compute_pos_and_vel`, `add_dipole` and `add_noise` #221
- **Breaking change**: add multiple TOD support to `describe_mpi_distribution` and make the field `MpiObservationDescr.tod_dtype` a list of strings #220
- Add links to the manual in the example notebook #219
- Implement new methods in the `Simulation` class: `set_scanning_strategy`, `set_instrument`, `set_hwp`, and deprecate `generate_spin2ecl_quaternions` #217
- Add `gzip_compression` keyword to `write_observations` #214

Simulation production

- See Giuseppe's talk
- First real application of the framework on a realistic case
- This activity is providing much visibility to the DMG and the Simulation Team

What is still missing

- HWP and proper 4π beam convolution (**extremely important!** work is ongoing)
- Dedicated map-making code for LiteBIRD (new working group has been established)
- Gain systematics & cross-correlations (work is ongoing)
- Time constant (work will start soon)
- Electronics systematics (detector non linearities)

The LiteBIRD Instrument Model

Purpose of the Instrument Model

- Description of the design of the instrument (number of detectors, placement, characteristics…)
- This is **extremely** important for the Simulation Team!
- Made of several parts:
  - Specifications (i.e., the *raw numbers*)
  - A database (the collection of raw numbers saved on a computer)
  - An interface to access the database

The story so far

- As seen from the outside, the LiteBIRD IMO team seems to have worked mostly to collect the raw numbers… 😕
- Currently they are being made available through wikipages (!)
- Each new version of the IMO gets a new page (potentially with different formatting and layout)
- The list of the wikipages (up to `v1.4`, which has not been released yet) is available here: LiteBIRD baseline configuration and parameters

Pages / ... / LiteBIRD baseline configuration and parameters Analytics

IMo-V1.2- July2021

Created by HENROT-VERSILLE Sophie, last modified on Oct 17, 2022

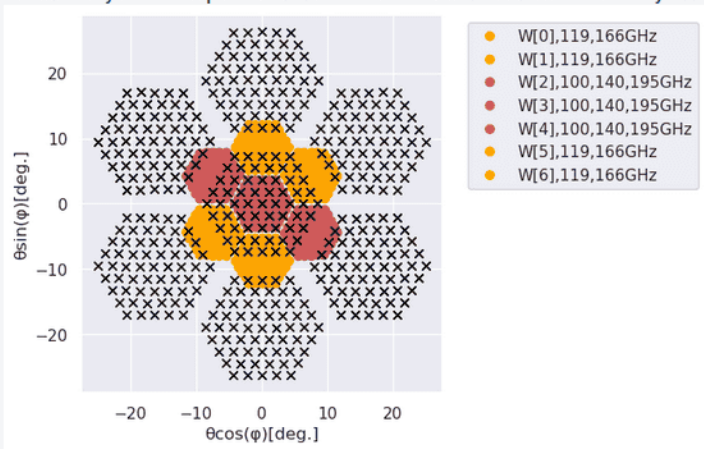
THIS IMo has never been delivered to the collaboration

work in progress...be patient...

NEW Updates with respect to IMo-V1.1

Leo Vacher and Yusuke Takase found two inconsistencies in the JSON file generated from IMoV1.1 see this document

- **one in the HFT pointing file: Cristian has reproduced it and the new one is linked below**
- **an inconsistency in the FOV reported in the IMO and the one simulated with TOAST to be exactly a factor of two larger - the text below is from Giuseppe:**



In the plot above : x refer to the pixel positions estimated from TOAST and orange and red points the nominal ones reported in the IMO

This bug is essentially due to the mis-interpretation of the info reported in the IMOv1.0 https://wiki.kek.jp/pages/viewpage.action?pageId=123338913

2. Beam sizes and detector sensitivities for the 15 bands

Sensitivity calculations (Takashi)

https://docs.google.com/spreadsheets/d/1dFGiC9bydefKYmuOvPHMKjrcmzb1y1pE_7hYDOp_6N4/edit#gid=202837976

details are given in Note # 65: LiteBIRD (Report of Sensitivity Task Force)

Take a New Screenshot

Freq [GHz]	Npix	Popt [pW]	NEPph [aW/rtHz]	NEPg [aW/rtHz]	NEPread [aW/rtHz]	NEPint [aW/rtHz]	NEPext [aW/rtHz]	NEPdet [aW/rtHz]	NETdet [microK.rtsec]	NETarr [microK.rtsec]
40	24	0.2918	5.444	3.948	2.674	7.237	4.094	8.315	114.63	18.50
60	24	0.2419	5.261	3.594	2.434	6.821	3.858	7.836	65.28	10.54
78	24	0.2686	5.978	3.787	2.565	7.527	4.258	8.648	58.61	9.46
50	12	0.3060	5.720	4.042	2.738	7.520	4.254	8.640	72.48	16.54
68	12	0.2709	5.810	3.804	2.576	7.407	4.190	8.510	68.81	15.70
89	12	0.2958	6.584	3.974	2.692	8.148	4.609	9.361	62.33	14.22
68	72	0.3279	6.576	4.184	2.834	8.294	4.692	9.529	105.64	9.84
89	72	0.3163	6.854	4.109	2.784	8.462	4.787	9.723	65.18	6.07
119	72	0.3765	8.181	4.484	3.037	9.811	5.550	11.272	40.78	3.80
78	72	0.3272	6.759	4.180	2.831	8.436	4.772	9.693	82.51	7.69
100	72	0.3061	6.970	4.043	2.738	8.510	4.814	9.778	54.88	5.11
140	72	0.3557	8.455	4.358	2.952	9.959	5.634	11.442	38.44	3.58
100	183	0.3559	7.619	4.359	2.953	9.262	5.239	10.641	71.70	4.19
119	244	0.4386	8.919	4.839	3.278	10.664	6.032	12.252	55.65	2.82
140	183	0.4206	9.269	4.739	3.210	10.894	6.162	12.516	54.00	3.16
166	244	0.3908	9.557	4.568	3.094	11.036	6.243	12.679	54.37	2.75
195	183	0.3572	9.784	4.367	2.958	11.115	6.288	12.770	59.61	3.48
195	127	0.6289	13.226	5.795	3.925	14.964	8.465	17.192	73.96	5.19
235	127	0.4708	12.315	5.014	3.396	13.723	7.763	15.767	76.06	5.34
280	127	0.3770	11.913	4.487	3.039	13.087	7.403	15.036	97.26	6.82
337	127	0.2997	11.582	4.000	2.710	12.549	7.099	14.418	154.64	10.85
402	169	0.2203	10.847	3.429	2.323	11.611	6.568	13.340	385.69	23.45

The need of a proper database

- A wikipage is a less-than-optimal solution:
  i. Hard to update, with many failure points (it happened...)
  ii. Tracking changes is hard
  iii. Numbers must be copied by hand in your calculator / spreadsheet / simulation code
  iv. Numbers must be updated by hand in simulation codes
- A proper database would be much better!

Implementing the database

- I (Maurizio Tomasi) started working on a database specification and implementation in September 2019, during the Santander F2F meeting
- The activity required to implement several things:
  i. A proper schema (= set of table specifications with relationships) for the database
  ii. An implementation of the schema, implemented using some SQL database
  iii. A web interface to the database
  iv. A way to bypass the database and get direct access to the raw numbers, for time-critical applications (e.g., the LiteBIRD Simulation Framework!)

InstrumentDB

- A mock version of InstrumentDB was presented at the Santander Meeting in September 2019
- After useful feedbacks from the IMO team, Luca Pagano, and Giuseppe Puglisi, in July 2020 I set up a repository: https://github.com/ziotom78/instrumentdb
- After several development versions (from 0.1.0 to 0.5.1), in January 2023 I released version 1.0.0 (Version 1.1.0 was released last week)
- The code is meant to be generic and can be used by other projects too; I branded it for LiteBIRD in the fork https://github.com/litebird/instrumentdb.

How does InstrumentDB work?

- The many parts of the instrument are organized in a hierarchical structure (tree):
  i. The three instruments `LFT`, `MFT`, and `HFT` are part of the `satellite`
  ii. Each detector is listed under a `frequency_channel`, like `L1-040`
  iii. Accessing information is possible through path-like strings, like `/satellite/LFT/L1-040/000_000_003_QA_040_T/detector_info`
- Each information can be updated (older versions do not disappear)
- Each information can be associated with a *specification document* (a Word file, a PDF, etc.)
- The database can be imported / exported to a directory tree

Live demo: https://litebirdimo.ssdc.asi.it

Using InstrumentDB for LiteBIRD (really!)

- Luca and Giuseppe have filled the database with IMO data so far
- Whoever wants to use the LiteBIRD Simulation Framework to run simulations must grab an exported copy of the IMO (web interface is too slow)
- The Simulation Framework accepts parameter files. Here is an example:

```
[simulation]
num_of_mc_runs = 500

[planet]
planet_name = "jupiter"
sed_file_name = "./jupiter_sed_ghz_k.csv"

[detector]
channel_obj = "/releases/v1.3/satellite/MFT/M2-166/channel_info"
sampling_rate_hz = 1.0
```

Instrument model objects

Data Files

Name	UUID	Upload date
channel_info	c9a0c4da-e512-4722-bf15-a486942ab79d	2020-10-09 17:08:20.555634+00:00

Source code used in the simulation

- Main repository: github.com/litebird/litebird_sim
- Version: 0.2.1, by The LiteBIRD simulation team

Report written on 2021-05-20 20:26:45

Updating the IMO using InstrumentDB (ideally)

1. Two database types are being used: the official *production database* (@SSDC) and *development databases* (which one can host on their own laptop)
2. A computer code creates a full description of the instrument and stores it in a local database hosted on the person's laptop
3. If everything looks fine, the person exports the database to a ZIP file, copies it on the SSDC server and imports it
4. After the production database has been updated, the SSDC exports the *entire* database to a ZIP file (containing *all* versions of the IMO, not just the latter) and makes it available to the collaboration

The role of SSDC

- The ASI SSDC agreed to host the IMO database on one of its servers
- The initial idea was to provide the SSDC with the source code of InstrumentDB and let them develop a more “professional” and solid site
- Because of various constraints, at the end we all agreed to keep using InstrumentDB. For this purpose, I created the “branded” fork mentioned before
- Gemma Luzzi is coordinating the activity of making the site publicly available, which currently involves Gianluca Polenta, Fabrizio Fabri, Daniele Navarra, and Antonio Guerra

What is still missing (1/2)

- The SSDC is settling out a few details about how to properly make the site visible to the outside world (A couple of days ago I sent them tips about how to configure logging and Apache)
- At the same time, the SSDC is looking for a way to streamline the possibility to perform regular updates of the software stack (adding features, security...), as at the moment everything is done by hand and is thus potentially error-prone
- Once online, it will be accessible at https://litebirdimo.ssdc.asi.it/
- This site should be the **only** proper way to publish and circulate the design of the instruments, sot it's important that the site is robust, secure, and featureful.

What is still missing (2/2)

- Apart from technical details, what we are still missing is a proper way for the IMO team to
  i. Collect information;
  ii. Release new versions of the IMO;
  iii. Validate the releases.
- So far, the process has been too chaotic and has caused failures
- It is important to **make the SSDC site accessible to the collaboration**, as the software stack could be a huge help to streamline IMO releases
- So far I did virtually all the coding for InstrumentDB. **I need somebody to help me in the future**!