

LumiBelle2: LGAD performance tracking

Pablo Mooney

CERN, Geneva, Switzerland

The University of Manchester, UK



Introduction

- We would like to monitor the performance of the LGADs over time (timeframe in the order of months)
- Look at the change in performance due to annealing
- We would like to monitor temperature/performance of the LGAD in the future
- Python processing pipeline now developed (V1)

File processing

- Process the daily simple_all_PV_1Hz root files
- Filter into stable windows
- Aggregate to a file/plot that spans over a few months
- Monitor the response of the LGAD against the ECL over time
- Include other information such as beam size

How it works

Takes a directory of root files, processes them into .csv files with stable windows tags and then plots across all the days

Can append any number of days at any time after processing

For example:

2026-04-21 to 2026-05-21 already processed, SCP copy next 3 days via VPN to Root Directory

Process just those three days and plot whole time frame with the three days added

```
2026-05-17 day already in aggregate
2026-05-18 day already in aggregate
2026-05-19 day already in aggregate
2026-05-20 day already in aggregate
2026-05-21 day already in aggregate
  Saving 2026-05-22 daily CSV...
  Saved daily CSV to: output/daily/2026-05-22.csv
  Saving 2026-05-22 to aggregate...
  Appended 2026-05-22 to aggregate (2755577 rows total)
  Saving 2026-05-23 daily CSV...
  Saved daily CSV to: output/daily/2026-05-23.csv
  Saving 2026-05-23 to aggregate...
  Appended 2026-05-23 to aggregate (2841617 rows total)
  Saving 2026-05-24 daily CSV...
  Saved daily CSV to: output/daily/2026-05-24.csv
  Saving 2026-05-24 to aggregate...
  Appended 2026-05-24 to aggregate (2927883 rows total)

— Aggregate summary:
Aggregate contains 34 file(s) from 2026-04-21 to 2026-05-24
Channel used: 2
```

How it works

Path to ROOT Directory



Run main()



Processes any new files



Appends them to aggregate CSVs per channel

Python Framework

- main.py
 - Has path to root files and output dirs
 - Select channel (1-4)
 - Options to wipe already processed data
 - Creates new aggregate csv per channel
 - Runs run() from run_analysis.py
- run_analysis.py
 - Set up process (omits already processed files/creates dir/wipes agg/daily)
 - Loads data using root_reader.py into a pandas data frame
 - stable_window_finder.py tags the data frame
 - Computes response and saves to CSVs using response.py and aggregate.py
 - Plots using analysis_plots.py

CSV

- Daily csv saves one files data into a csv (toggled by SAVE_DAILY)
- aggregated.csv is all the daily files. Each file processed adds to the end of this

Structure of CSV:

- Timestamp (JST)
- Beam parameters: ECL, INJ_BEAM_STATUS, SIGMAYatIP, EMITTY, TIL
- is_stable, is_very_stable tag
- For stable windows: response and time window
- Source date (one day corresponds to 8am-8am 24hr period)

CSV example

timestamp	ECL	SKB_CGHINJ_BEAM_GATE_STATUS	SKB_CGLINJ_BEAM_GATE_STATUS	SKB_BMHXR_M_BEAM_SIG_MAYatIP	SKB_BMLXR_M_BEAM_SIG_MAYatIP	SKB_BMHXRM_BEAM_EMITT_Y	SKB_BMLXRM_BEAM_EMITT_Y	TIL	is_stable	is_very_stable	t_start	t_end	response	source_date
2026-05-03 08:11:29.34270 6203+09:00	40339.6 3267991 6205	0	0	269.450104 89706766	229.189152 8654429	80.67039892 286783	58.36407532 3452745	1931.05411	TRUE	FALSE	2026-05-03 08:11:01.30 2729607+09 :00	2026-05-03 08:12:00.389 226198+09:0 0	0.0478699	03.05.2026
2026-05-03 08:11:30.34428 8111+09:00	42250.8 451	0	0	269.450104 89706766	229.189152 8654429	80.67039892 286783	58.36407532 3452745	1937.33941	TRUE	FALSE	2026-05-03 08:11:01.30 2729607+09 :00	2026-05-03 08:12:00.389 226198+09:0 0	0.04585327	03.05.2026
2026-05-03 08:11:29.34270 6203+09:00	40339.6 3267991 6205	0	0	269.450104 89706766	229.189152 8654429	80.67039892 286783	58.36407532 3452745	1931.05411	TRUE	FALSE	2026-05-03 08:11:01.30 2729607+09 :00	2026-05-03 08:12:00.389 226198+09:0 0	0.0478699	03.05.2026

main.py

- File path: contains all the root files, can add a new file per day and it will only process that one file if the others are already processed
- Aggregate is the csv with all the days data processed so far (including stable windows)
- Save daily toggle

```
FILE_PATH      = Path("../RootFiles")
DAILY_DIR      = Path("output/daily")
AGGREGATE_DIR  = Path("output/aggregate")

CHANNEL        = 2
WIPE_AGGREGATE = False
WIPE_DAILY     = False
SAVE_DAILY     = True

PLOT_ANALYSIS  = True
PRINT_STATS    = True
VERBOSE        = True

run(
    file_path      = FILE_PATH,
    daily_dir      = DAILY_DIR,
    aggregate_dir  = AGGREGATE_DIR,
    channel        = CHANNEL,
    wipe_aggregate = WIPE_AGGREGATE,
    plot_analysis  = PLOT_ANALYSIS,
    print_stats_flag = PRINT_STATS,
    verbose        = VERBOSE,
    save           = SAVE_DAILY,
    wipe_daily_flag = WIPE_DAILY
)
```

Root reader

- Currently reads in these branches
- Can add/remove to plot different parameters / improve processing time
- Also

til_branch =

f"B2LUMI_LUMIBELLE2_CH{channel}_TIL4"

```
BRANCHES = [  
    "TIMESTAMP",  
    "B2_nsm_get_ECL_LUM_MON_lum_accel",  
    "SKB_CGHINJ_BEAM_GATE_STATUS",  
    "SKB_CGLINJ_BEAM_GATE_STATUS",  
    "SKB_BMHXRM_BEAM_SIGMAYatIP",  
    "SKB_BMLXRM_BEAM_SIGMAYatIP",  
    "SKB_BMHXRM_BEAM_EMITTY",  
    "SKB_BMLXRM_BEAM_EMITTY",  
]
```

Stable Windows

- Based on Anass' method
- Tags the csvs with `is_stable` / `is_very_stable` rather than duplicating data
- Can change parameters to adjust sensitivity on `stable_window_finder.py`
- Response only calculated across windows



Response

ECL/TIL is calculated per stable window

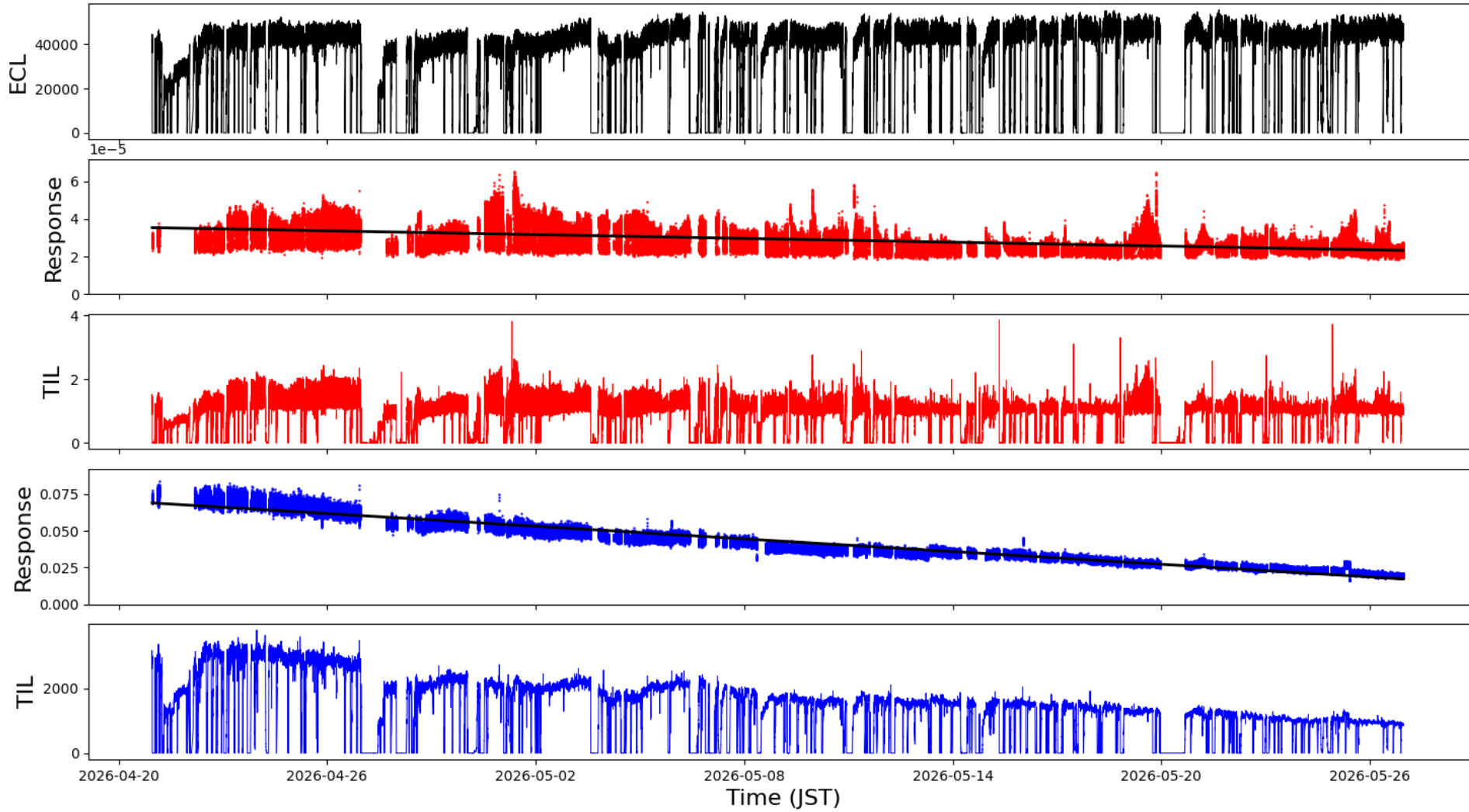
- This is stored in CSV
- Non stable windows are recorded as NAN

Plot function

- Can now plot cross channels, any combination of parameters
- Linear regression test for response

Plot example

CVD



LGAD
25pix



Next Steps

- Need to clean up plotting function
- Write instructions/README
- Available on GIT