IV curves update

Anna Balboni, Alessandro Minotti

NP04 PDS Data Taking Planning Meeting

30/05/2024

INFN FERRARA
Istituto Nazionale di Fisica Nucleare
Sezione di Ferrara

Università degli Studi di Ferrara

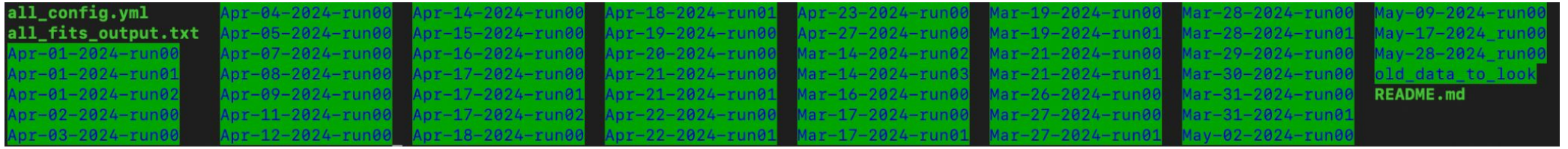
Content

- New file organization
- IV analysis algorithm
- New scripts for Vbd monitoring
- Vbd results from 19th April to 28th May
- Next steps

New file organization

- Acquired IV curves (now taken weekly) in:

  /eos/experiment/neutplatform/protodune/experiments/ProtoDUNE-II/PDS_Commissioning/ivcurves



- All scripts regarding IV curves moved to:

  https://github.com/DUNE/PDS/tree/main/scripts/iv_analysis

- Results of IV analysis are in:

  https://github.com/DUNE/PDS/tree/main/data/iv_analysis

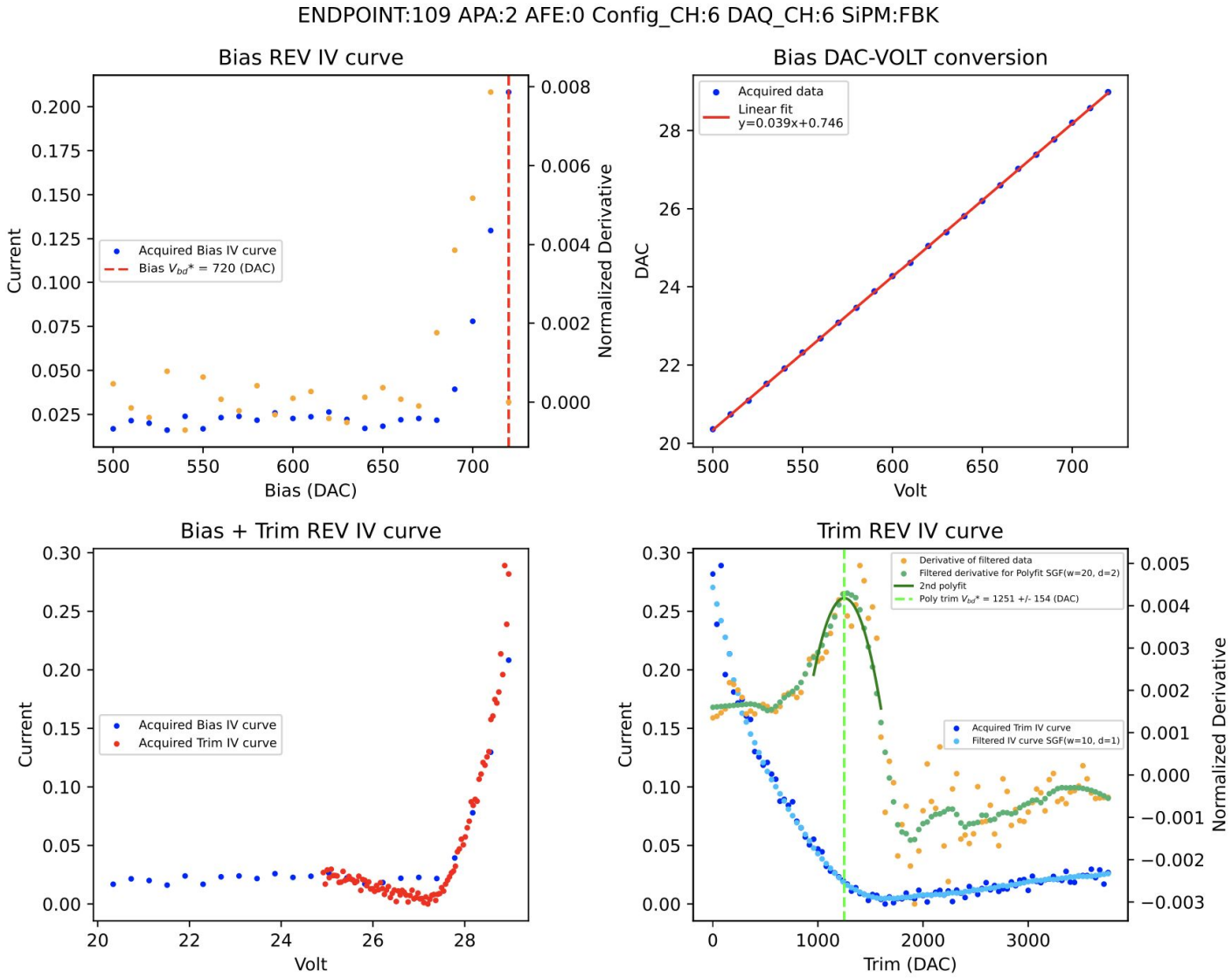
- Json maps for IV curve acquisition (with/without dead channels) are in:

  https://github.com/DUNE/PDS/tree/main/maps

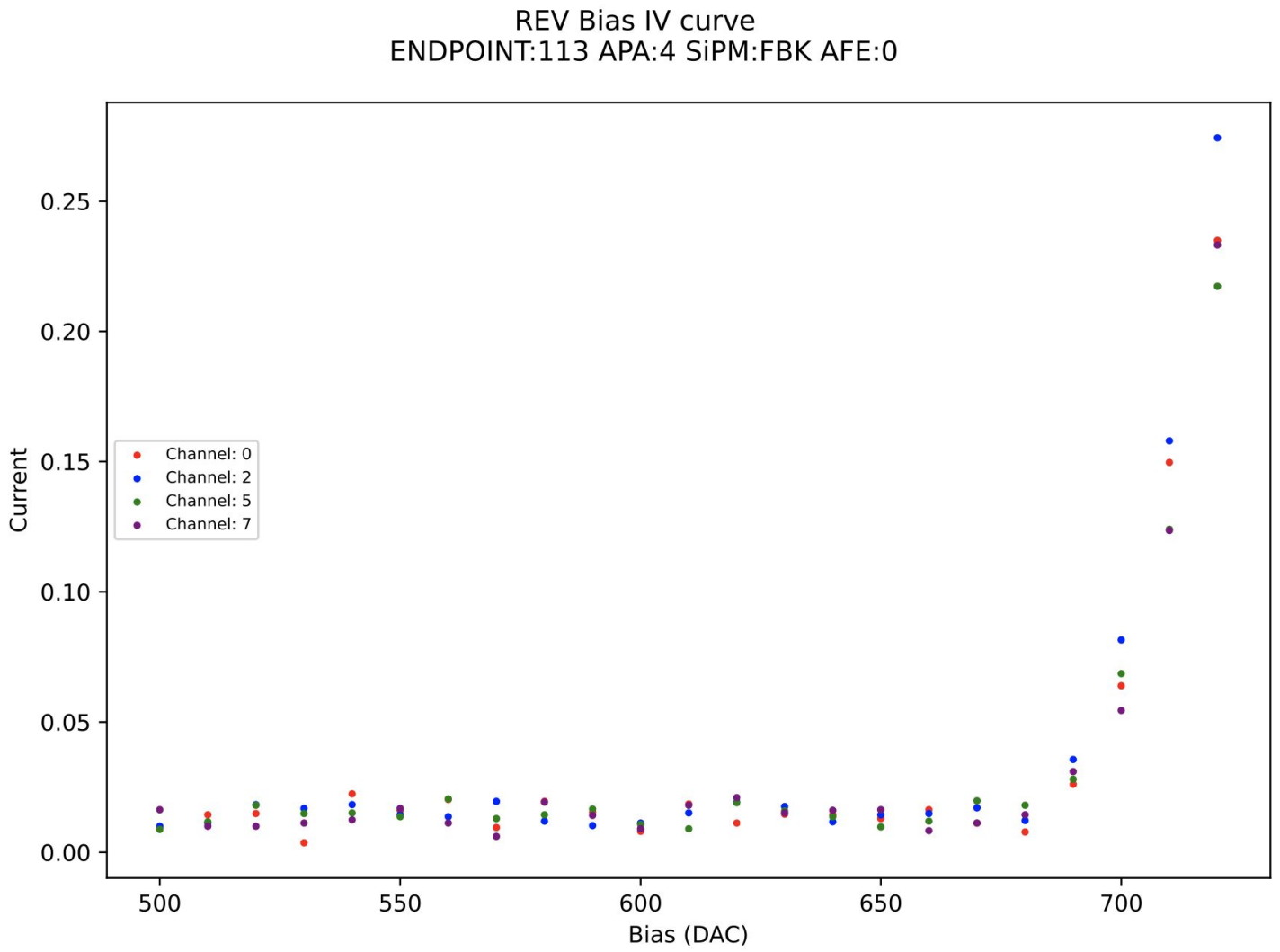
IV curve algorithm

- The program for IV analysis is called: **IV_analysis.py** + IV_analysis_utils.py (useful libraries and functions).
- It determines the **breakdown voltage** and estimates the **operation voltage**.
- It requires the following input parameters:
  - input_dir : /eos/experiment/neutplatform/protodune/experiments/ProtoDUNE-II/PDS_Commissioning/ivcurves/YYYYYY (related to a given run)
  - output_dir : https://github.com/DUNE/PDS/tree/main/data/iv_analysis (default)
  - endpoint : 104, 105, 107, 109, 111, 112, 113 or ALL
  - trimfit : poly (default) , pulse or both
  - map_path : /afs/cern.ch/user/a/anbalbon/IV_curve/PDS/maps/original_channel_map.json (default)
- It produces four output files (for each endpoint, of a given run):
  - XX.XX.XXX.XXX_plots.pdf
  - XX.XX.XXX.XXX_output.txt
  - XX.XX.XXX.XXX_Bias_IVplots_AFE.pdf
  - XX.XX.XXX.XXX_dic.json

XX.XX.XXX.XXX_plots.pdf



XX.XX.XXX.XXX_Bias_IVplots_AFE.pdf



XX.XX.XXX.XXX_output.txt

IP	File_name	APA	AFE	Config_CH	DAQ_CH	SIPM_type	Run	Endpoint_timestamp	Start_time	End_time	Bias_data_quality	Bias_min_I
Bias_max_I	Vbd_bias(DAC)	Vbd_bias(V)	Vbd_bias_error(V)	Bias_conversion_slope	Bias_conversion_intercept	Trim_data_quality	Trim_min_I	Trim_max_I				
Fit_status	Poly_Vbd_trim(DAC)	Poly_Vbd_trim_error(DAC)	Pulse_Vbd_trim(DAC)	Pulse_Vbd_trim_error(DAC)	Vbd(V)	Vbd_error(V)						
10.73.137.113	apa_4_afe_0_ch_0.root	4	0	0	0	FBK	May-09-2024-run00	May-09-2024_2153	May-09-2024_2154	May-09-2024_2157	Good	
0.004	0.235	720	28.666	0.038	0.03916	0.469	Good	-0.097	0.239	Only polyfit	1078	129
nan	nan	27.508	0.144									
10.73.137.113	apa_4_afe_0_ch_2.root	4	0	2	2	FBK	May-09-2024-run00	May-09-2024_2153	May-09-2024_2157	May-09-2024_2201	Good	
0.010	0.274	720	28.667	0.039	0.03916	0.471	Good	-0.093	0.246	Only polyfit	1229	128
nan	nan	27.346	0.143									
10.73.137.113	apa_4_afe_0_ch_5.root	4	0	5	5	FBK	May-09-2024-run00	May-09-2024_2153	May-09-2024_2201	May-09-2024_2205	Good	
0.009	0.217	720	28.656	0.028	0.03915	0.470	Good	-0.094	0.243	Only polyfit	1093	87
nan	nan	27.481	0.098									
10.73.137.113	apa_4_afe_0_ch_7.root	4	0	7	7	FBK	May-09-2024-run00	May-09-2024_2153	May-09-2024_2205	May-09-2024_2208	Good	
0.006	0.233	720	28.663	0.031	0.03915	0.472	Good	-0.095	0.233	Only polyfit	1198	163
nan	nan	27.375	0.178									

XX.XX.XXX.XXX__dic.json

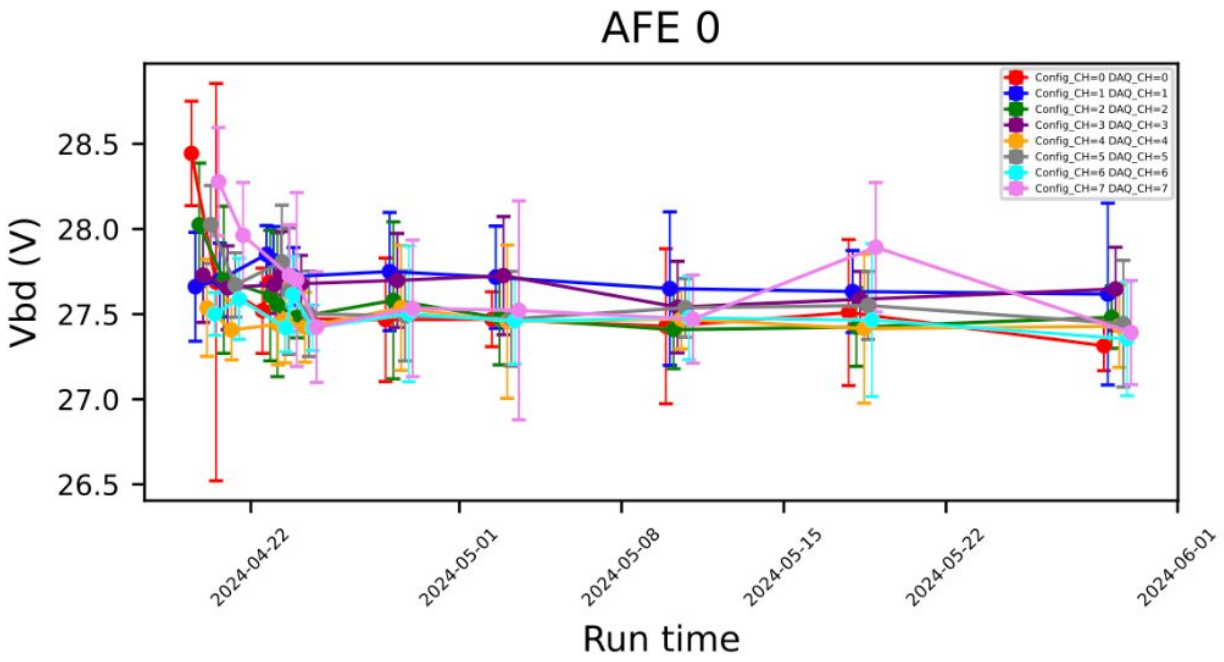
- root
  - apa 1
  - fbk [] 8 items
    - 0 0
    - 1 1
    - 2 2
    - 3 3
    - 4 4
    - 5 5
    - 6 6
    - 7 7
  - hpk [] 8 items
    - 0 8
    - 1 9
    - 2 10
    - 3 11
    - 4 12
    - 5 13
    - 6 14
    - 7 15
  - FBK_op_bias [] 1 item
    - 0 782
  - FBK_op_trim [] 8 items
    - 0 267
    - 1 61
    - 2 286
    - 3 163
    - 4 227
    - 5 167
    - 6 220
    - 7 228
  - HPK_op_bias [] 1 item
    - 0 1135
  - HPK_op_trim [] 8 items
    - 0 140
    - 1 167
    - 2 170
    - 3 239
    - 4 101
    - 5 93
    - 6 55
    - 7 141
  - timestamp "May-09-2024_2153"
  - run "May-09-2024-run00"
  - ip "10.73.137.104"

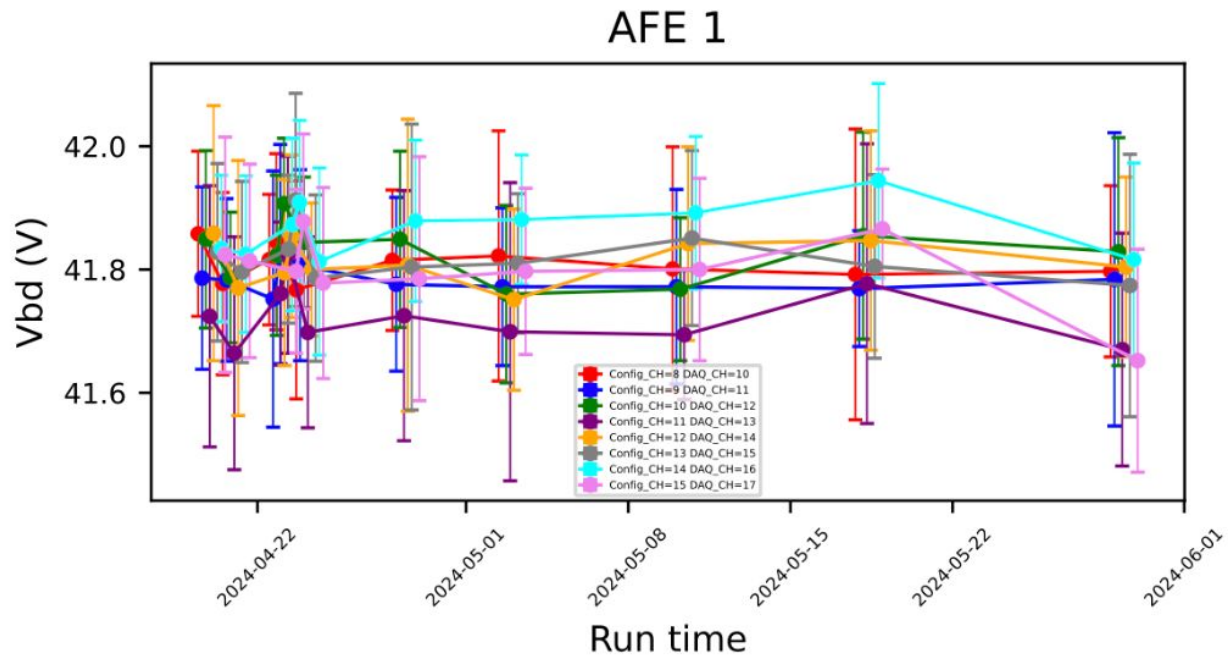
New scripts for Vbd monitoring

- **Vbd_plot_single_run.py** + Vbd_plot_utils.py (useful libraries and functions) which allows to monitor the Vbd of a single run and produces two different plots:
  - VB_HIST_X_RUN → an histogram with all endpoints (divided in FBK and HPK)
  - CH_VBD_X_RUN → a plot where on the y axis there is the Vb of each channel and on the x axis the name of the corresponding daq channel (all CH in the same plot)

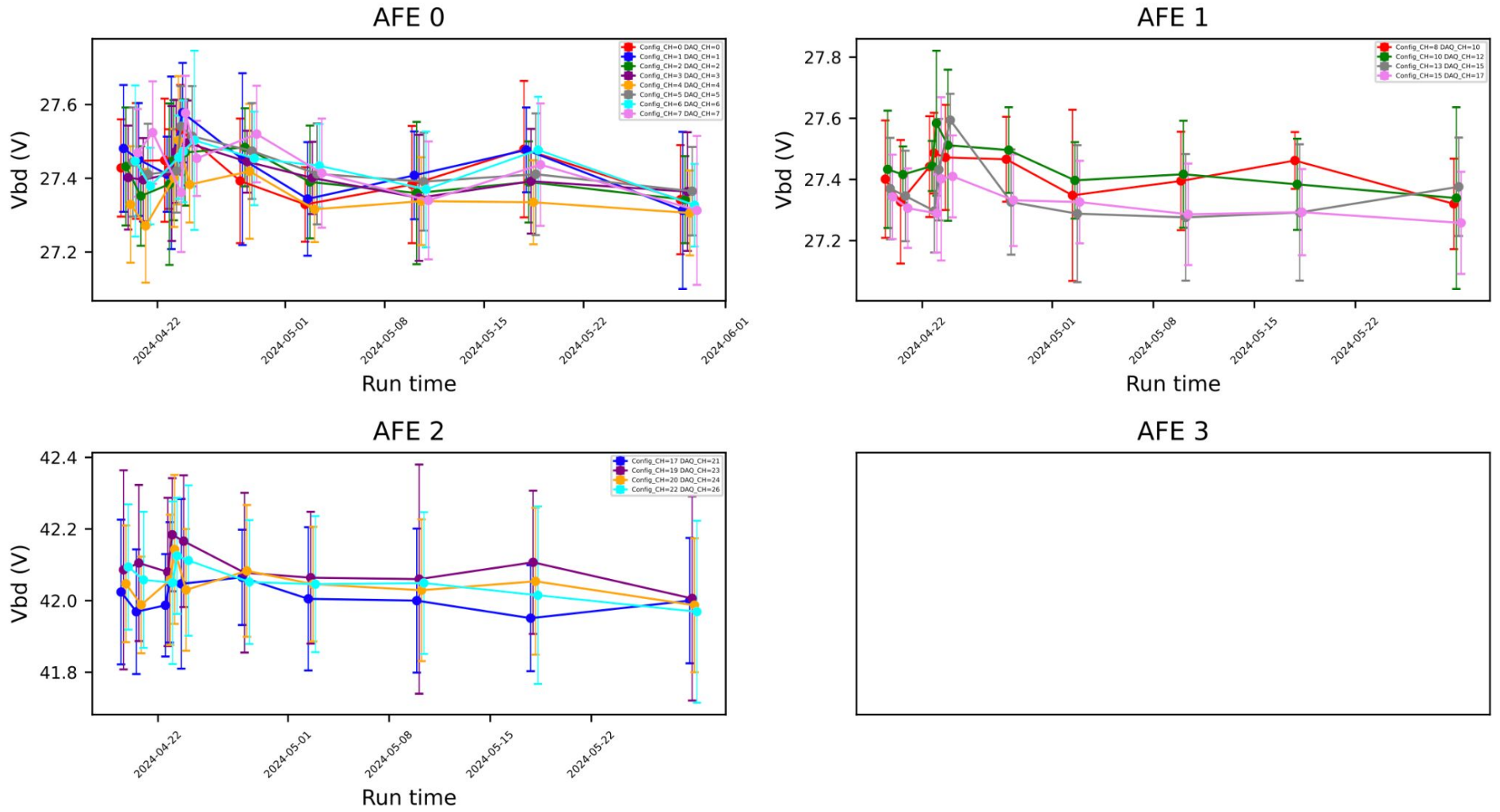
- **Vbd_plot_all_run.py** + Vbd_plot_utils.py (useful libraries and functions) which allows to see how Vbd varies in time and produces two different plots:
  - CH_VBD_VS_RUN → a plot for each AFE with the channel Vbd as a function of time
  - AFE_VBD_VS_RUN → a plot for each endpoint with mean AFE Vbd as a function of time

Example: CH_VBD_VS_RUN for endpoint 104

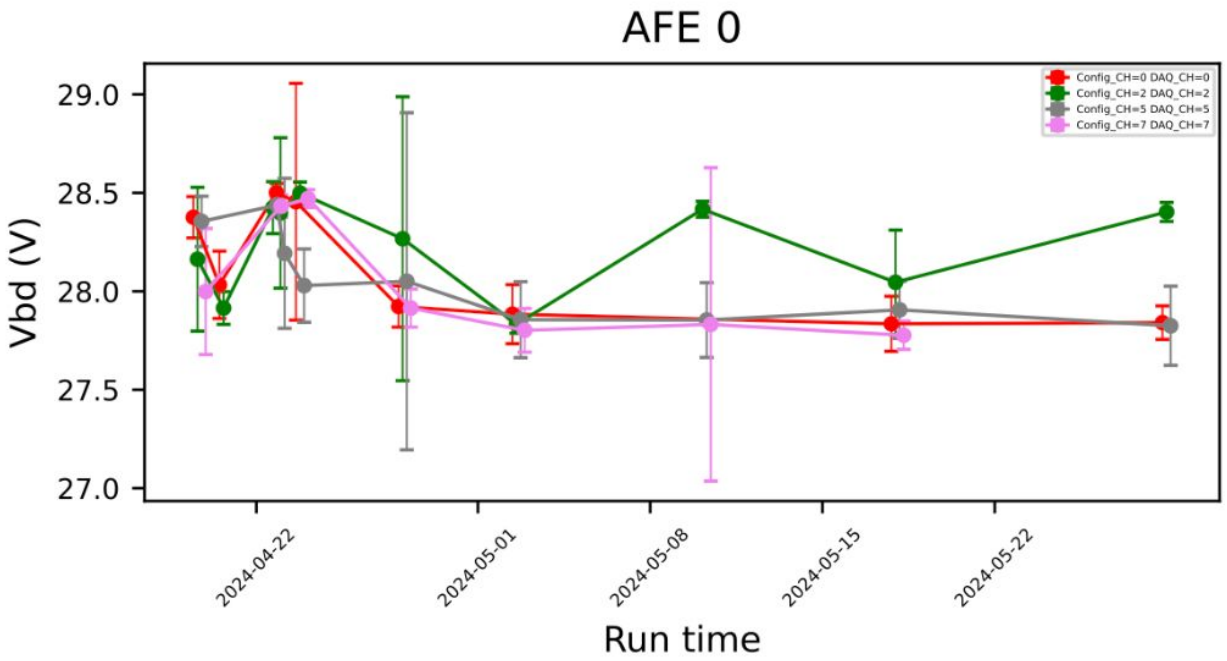


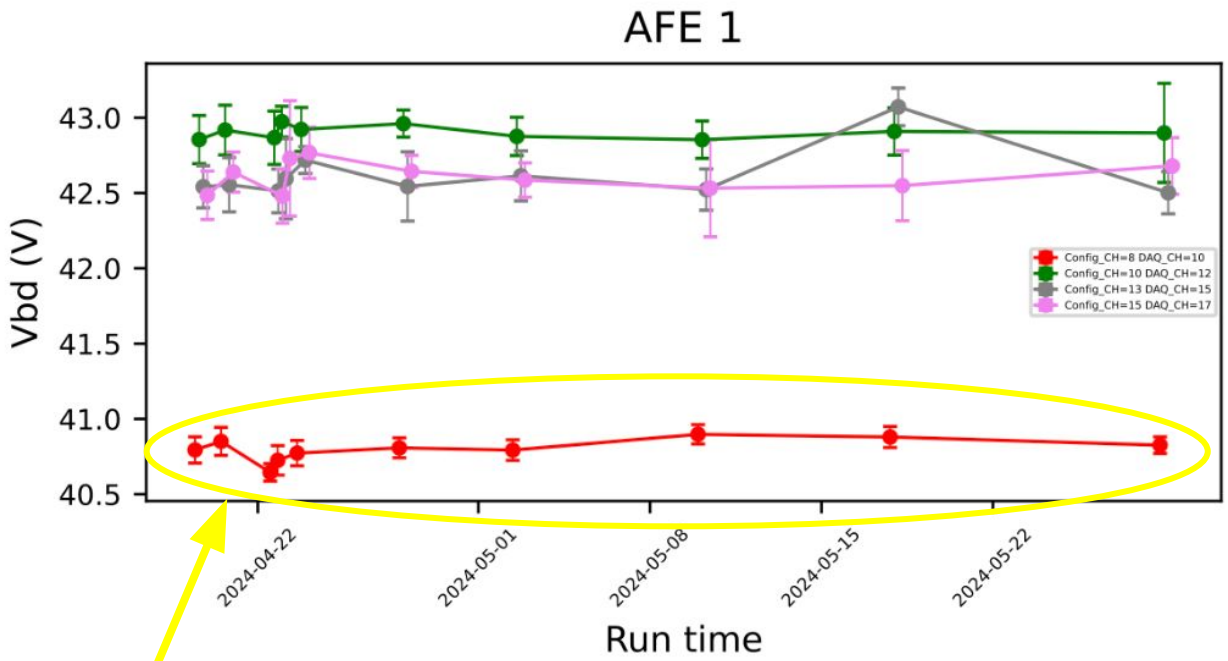


Example: CH_VBD_VS_RUN for endpoint 105



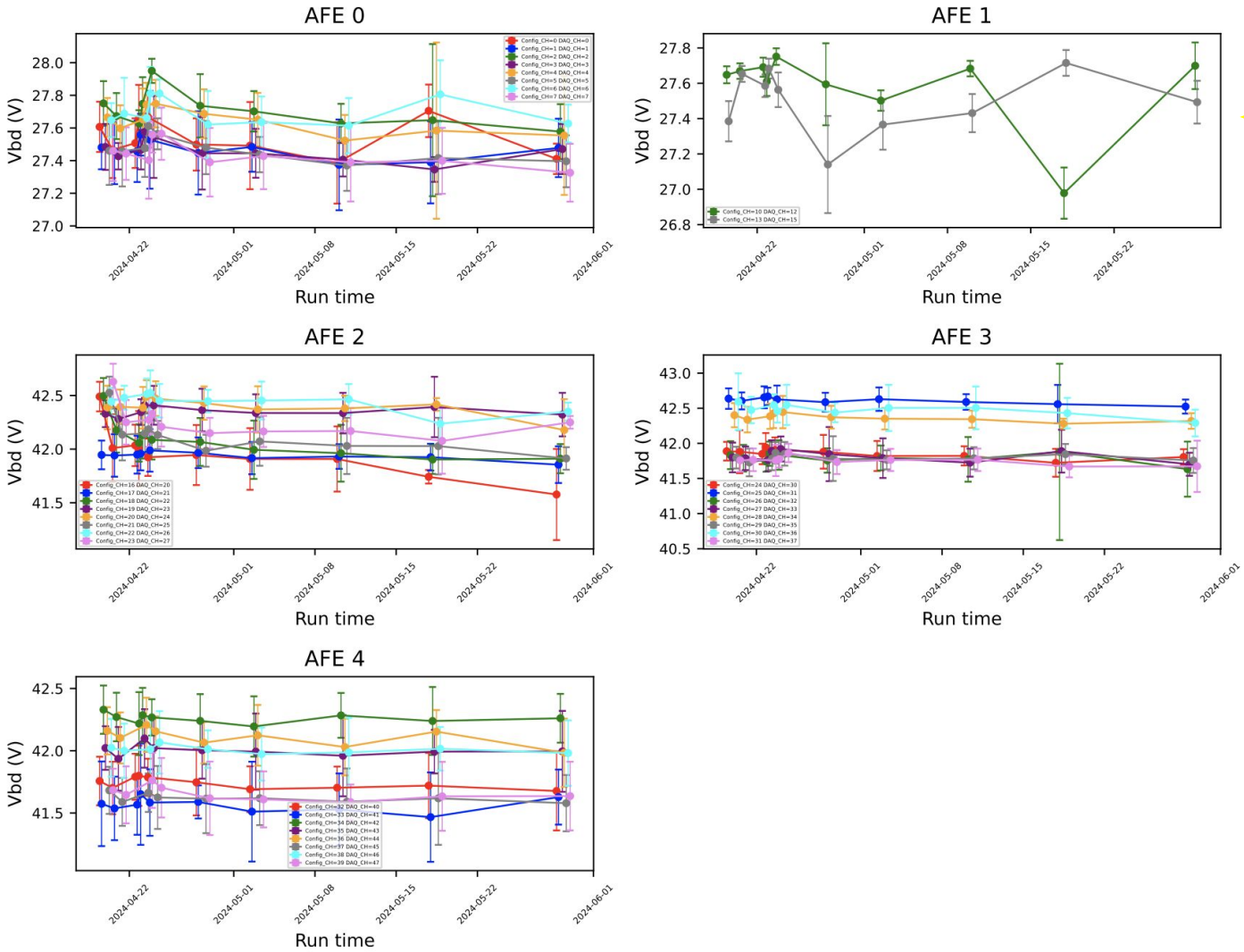
Example: CH_VBD_VS_RUN for endpoint 107





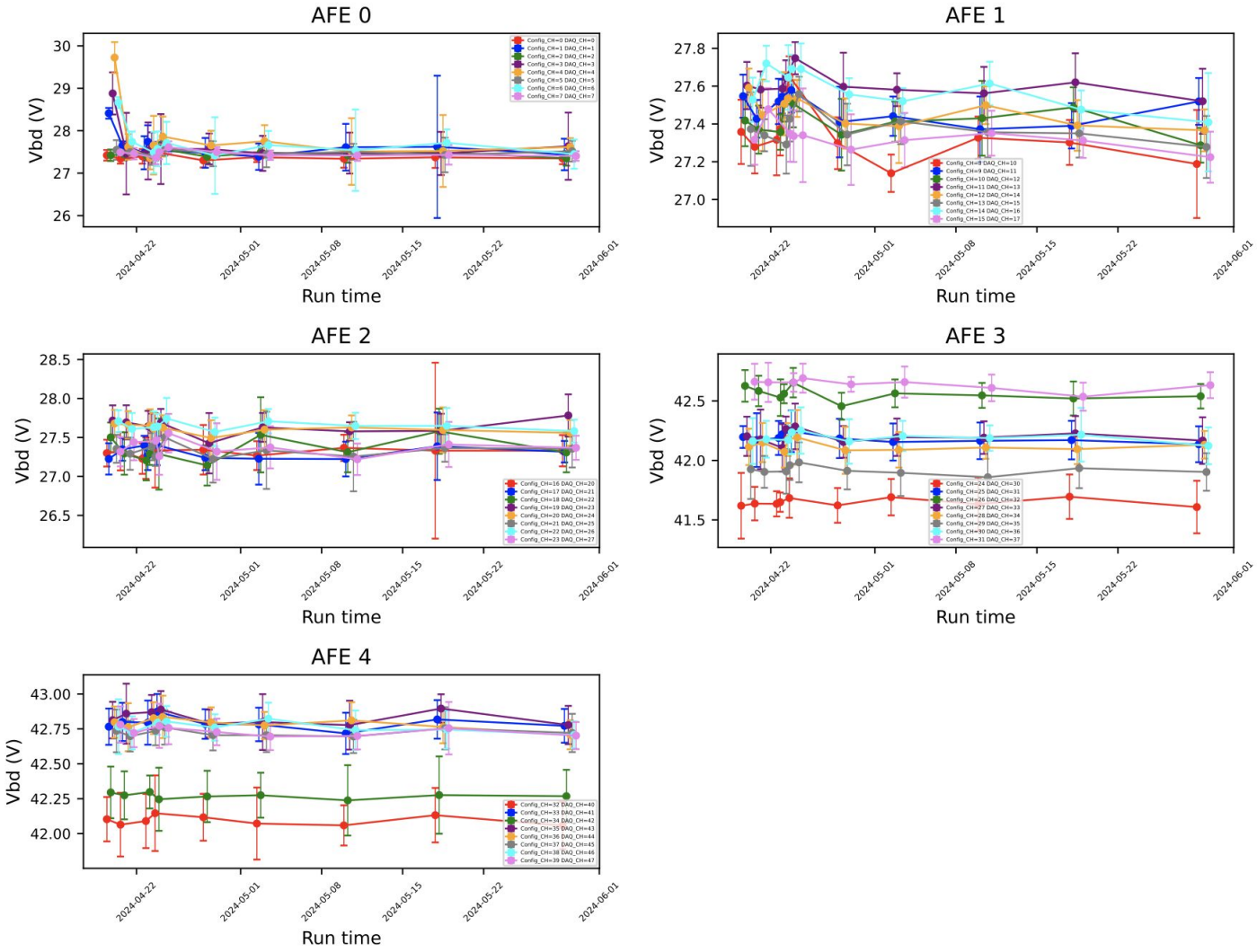
Config_CH: 8 DAQ_CH: 10
HAS A LOWER VBD

Example: CH_VBD_VS_RUN for endpoint 109

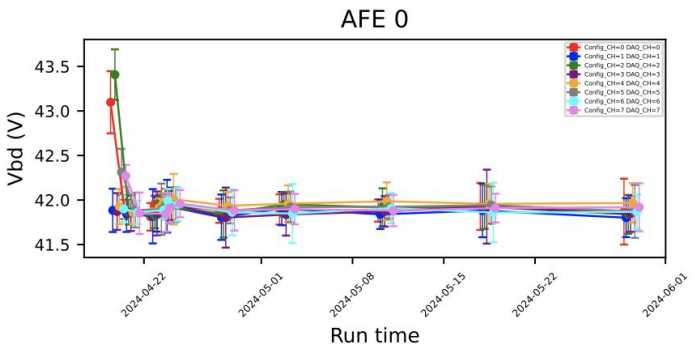


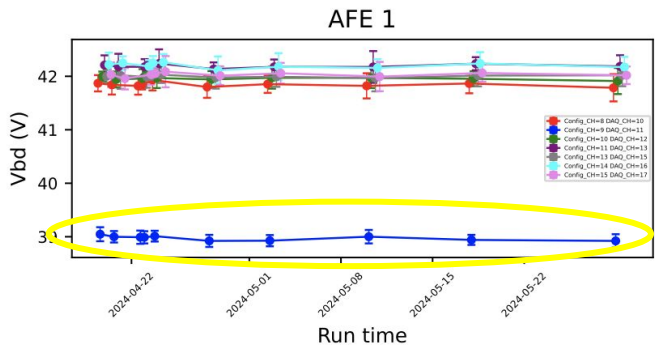
FOUR CHANNELS ARE MISSING

Example: CH_VBD_VS_RUN for endpoint 111

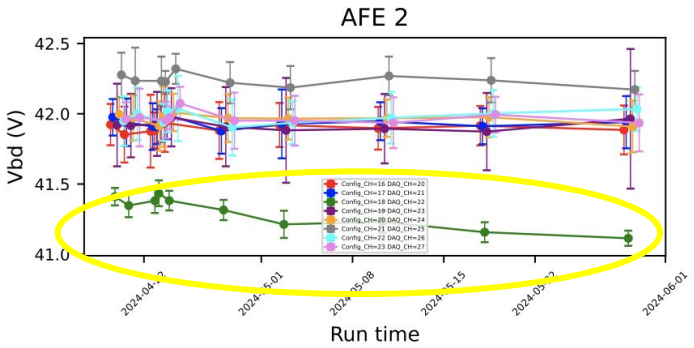


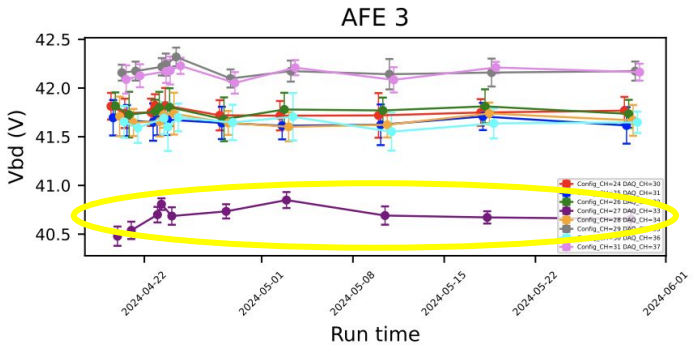
Example: CH_VBD_VS_RUN for endpoint 112

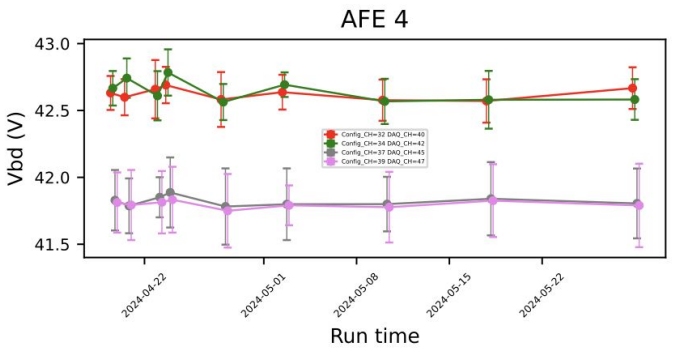




ONE CHANNEL IS MISSING

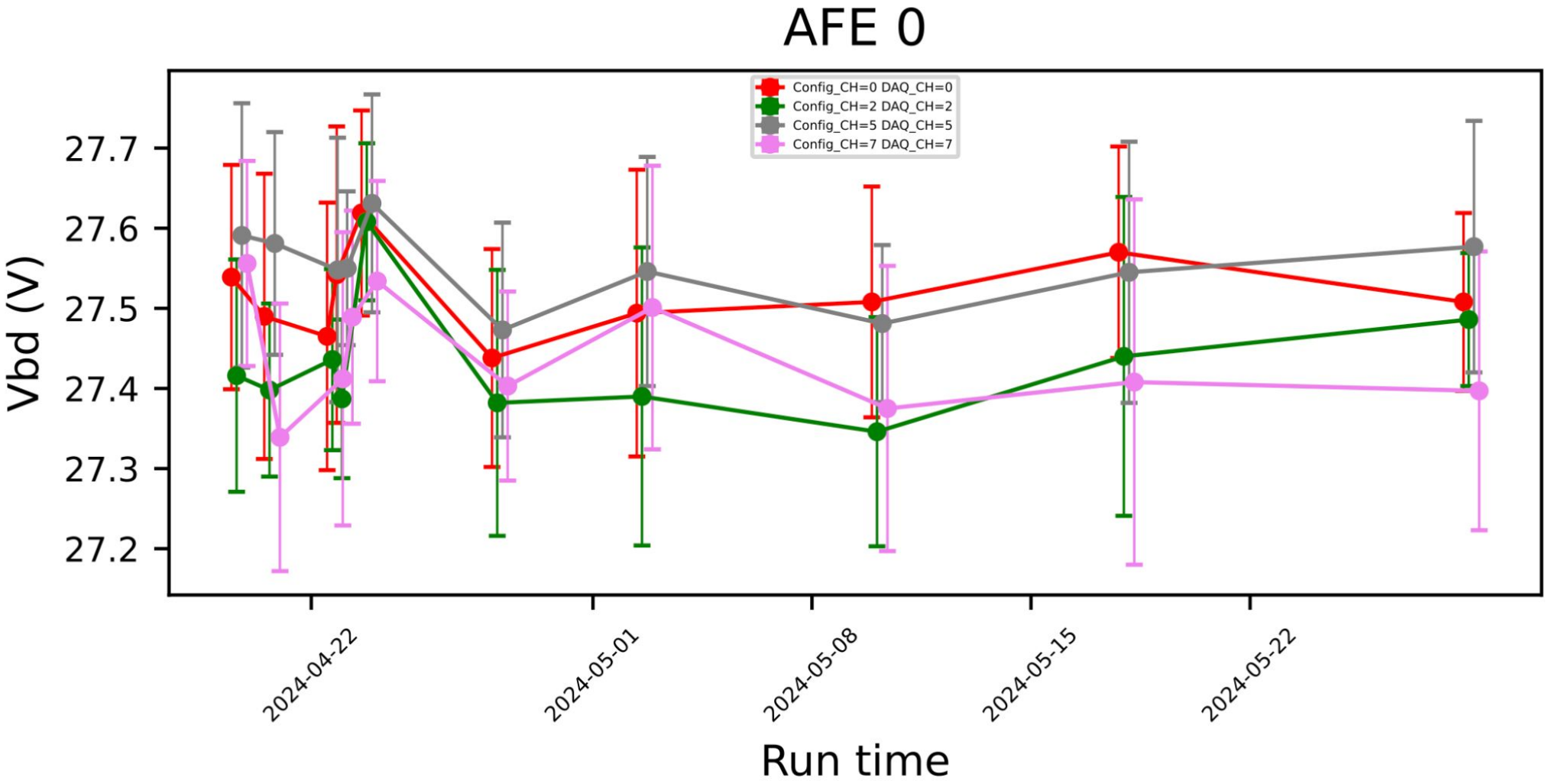




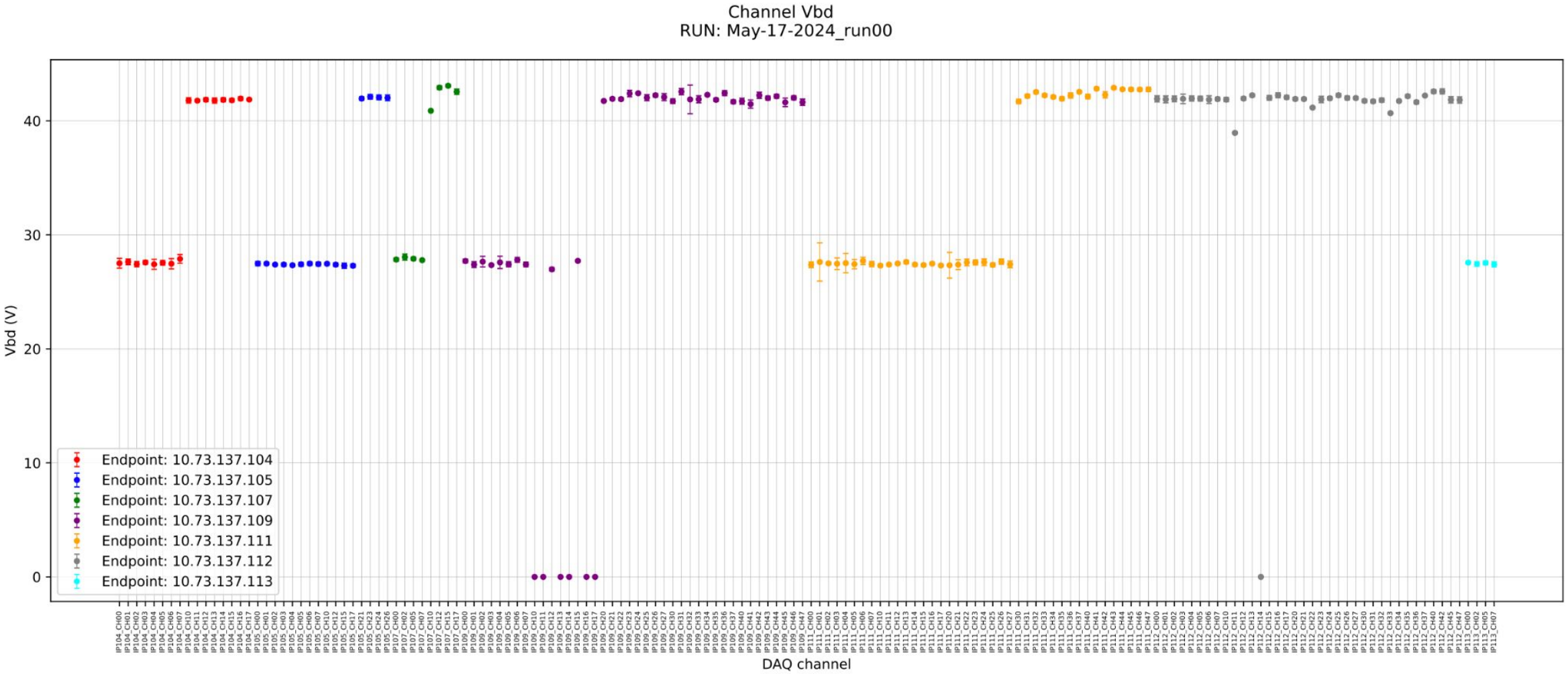


THREE CHANNELS HAVE A LOWER VDB

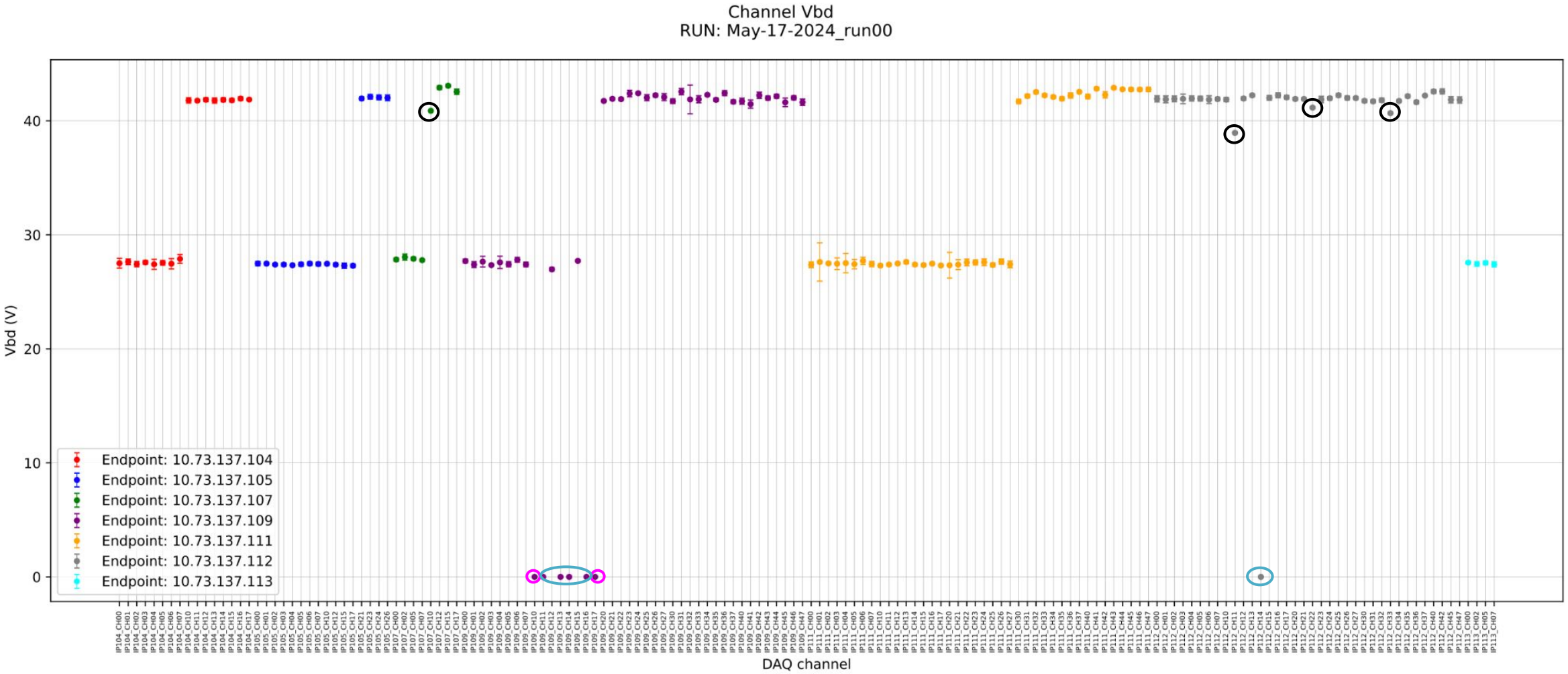
Example: CH_VBD_VS_RUN for endpoint 113



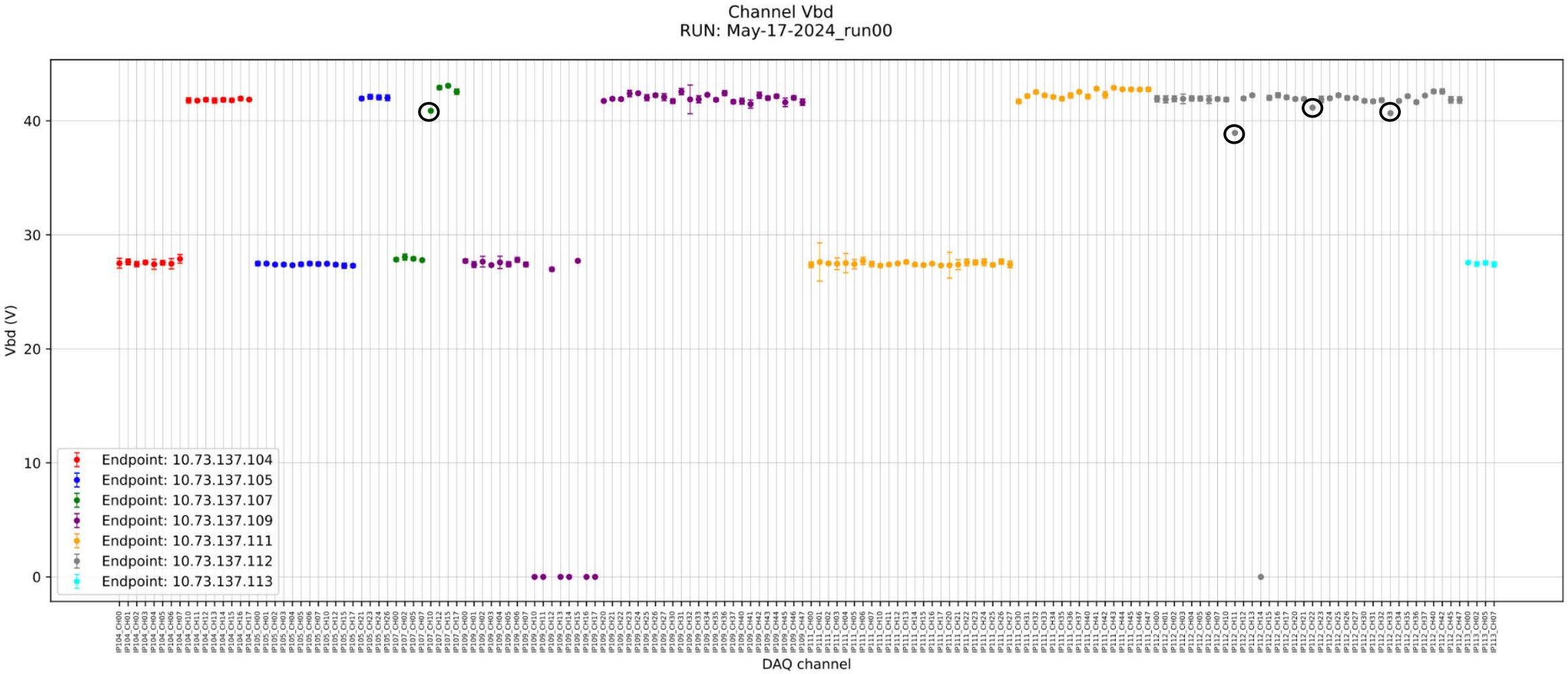
Example: CH_VBD_X_RUN for May-17-2024 run



We noticed some strange Vbd values

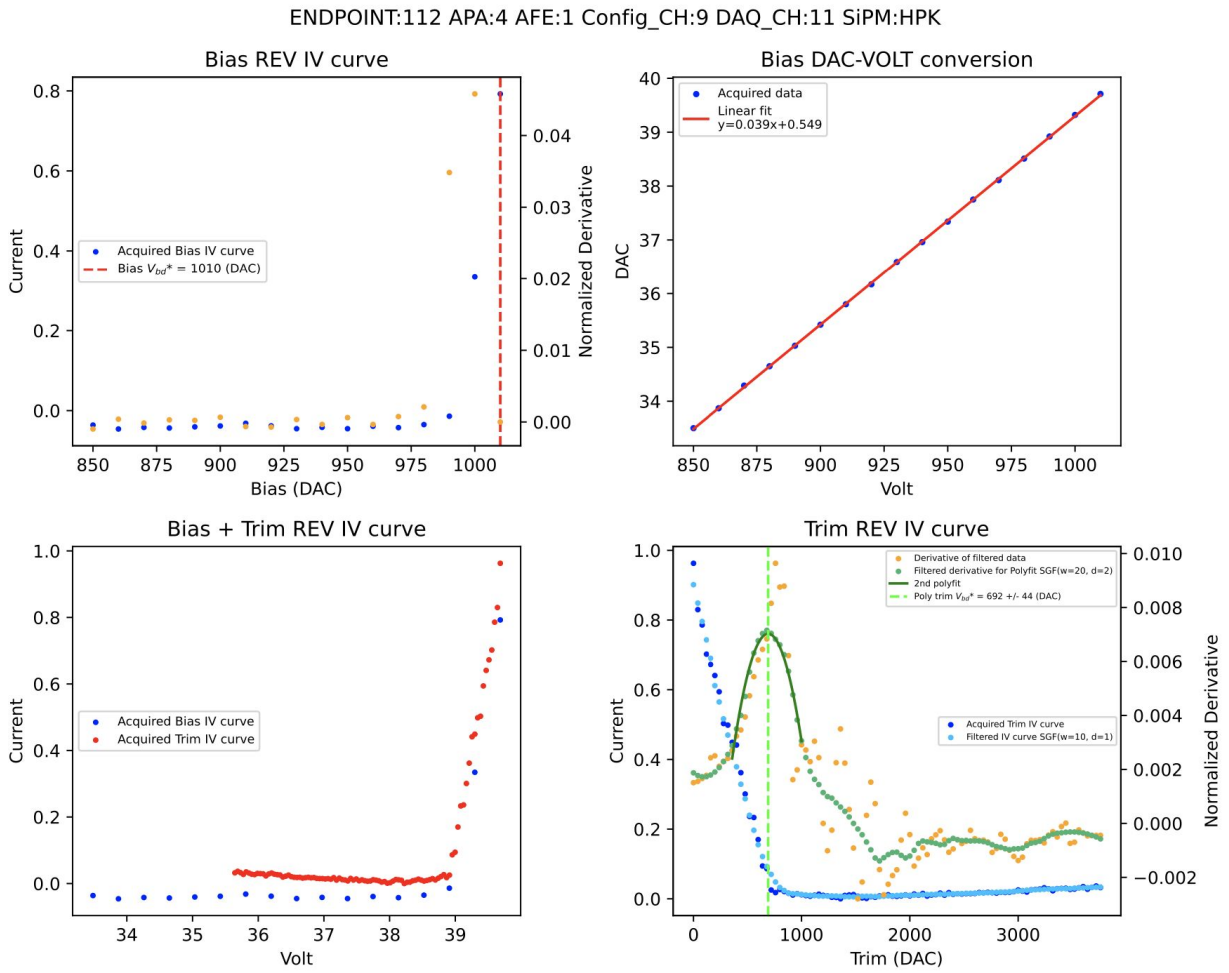


Regarding channels with low Vbd…

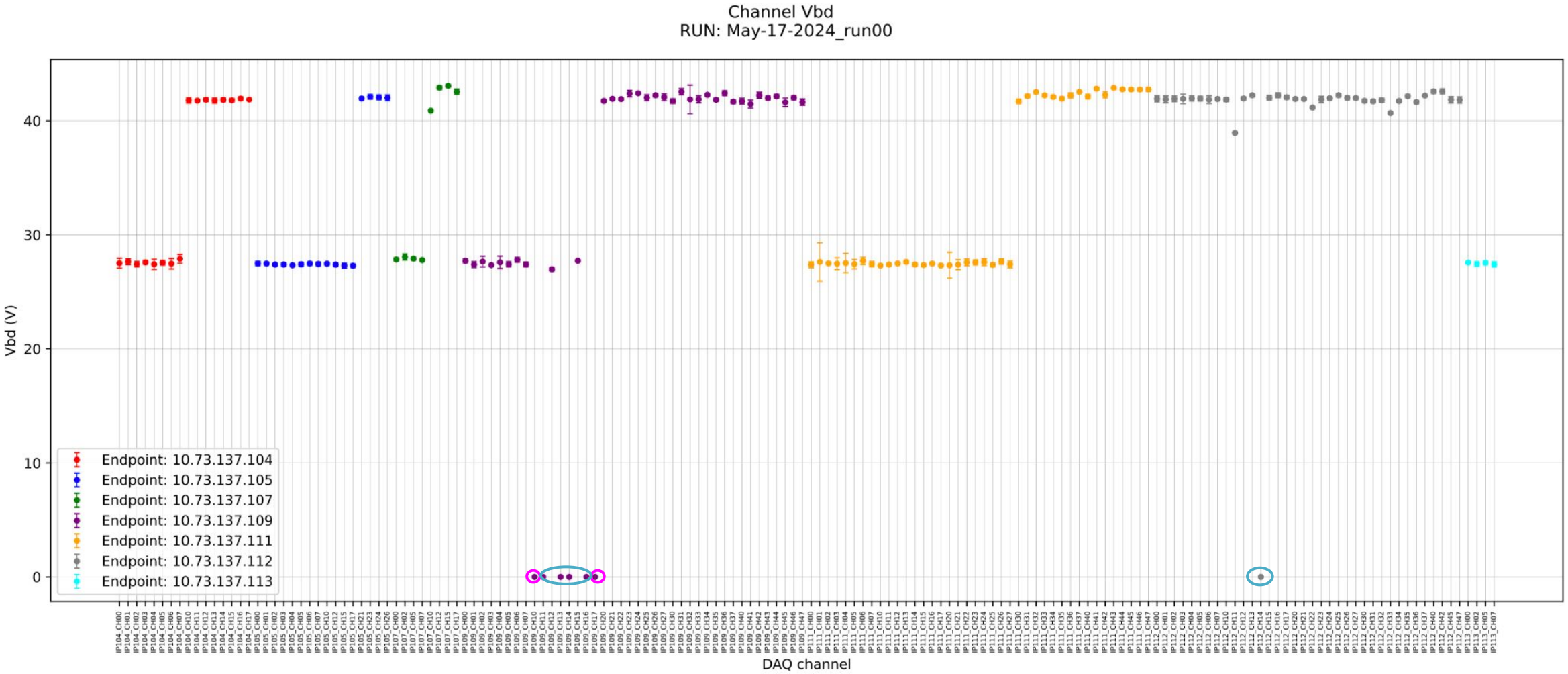


We noticed that these channels have a “*steep IV curve*”:

- DAQ_CH 10 - endpoint 107
- DAQ_CH 11 - endpoint 112
- DAQ_CH 22 - endpoint 112
- DAQ_CH 33 - endpoint 112



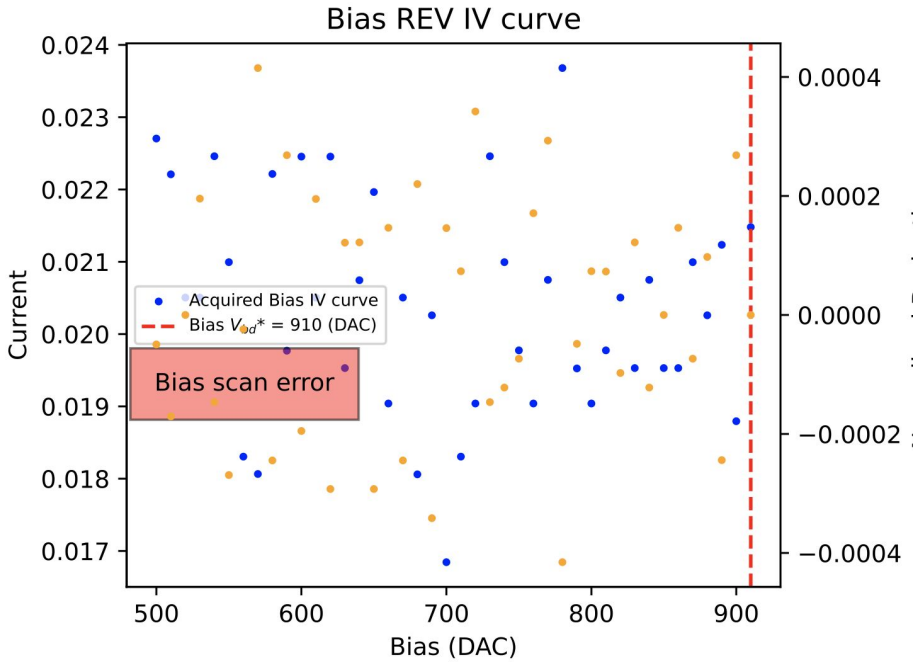
Regarding channels with Vbd set to zero…



We noticed that:

- DAQ_CH 10 - endpoint 109
- DAQ_CH 17 - endpoint 109

have a bad bias IV curve



- DAQ_CH 11 - endpoint 109
- DAQ_CH 13 - endpoint 109
- DAQ_CH 14 - endpoint 109
- DAQ_CH 16 - endpoint 109
- DAQ_CH 14 - endpoint 112

IV_curve file.root is missing

Information about dead and strangly-behaving channels is stored in the [mapping spreadsheet]

Next steps

- Add all scripts / data on GitHub (PDS)
- Produce documentation in GitHub (the scripts are already documented in the experts documentation)
- Improve the algorithm for analyzing IV curves to handle noisy channels
- Produce an update json map with only “good channels” (not-dead / only connected ones) and use it to acquire IV curves
- Produce a new json map that will be used to set the operation bias, with the following organization:

```
'10.73.137.104': {'apa': 1, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.105': {'apa': 1, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.107': {'apa': 1, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.109': {'apa': 2, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.111': {'apa': 3, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.112': {'apa': 4, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]},
'10.73.137.113': {'apa': 4, 'Vop_bias': [ , , , , ], 'Vop_trim': [ , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , , ]}
```

where Vop_bias is an array of 5 elements (one for each AFE) and Vop_trim is an array of 40 elements (one for each CH)

Next steps

- Take data with updated maps as input
- Equalize the gains for self-trigger channels and cross check with the IV
- Equalize the full stream channels (less trivial)

Thank you for the attention!