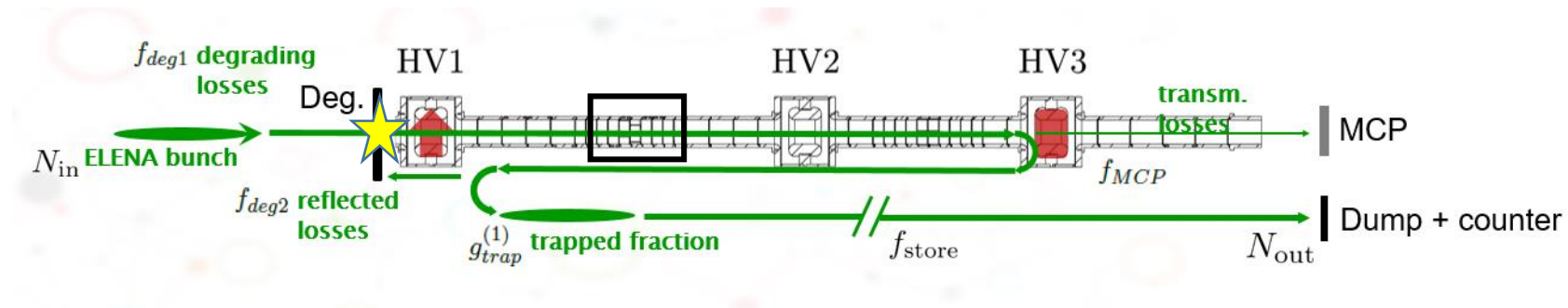


# Efficient Accumulation of Antiprotons

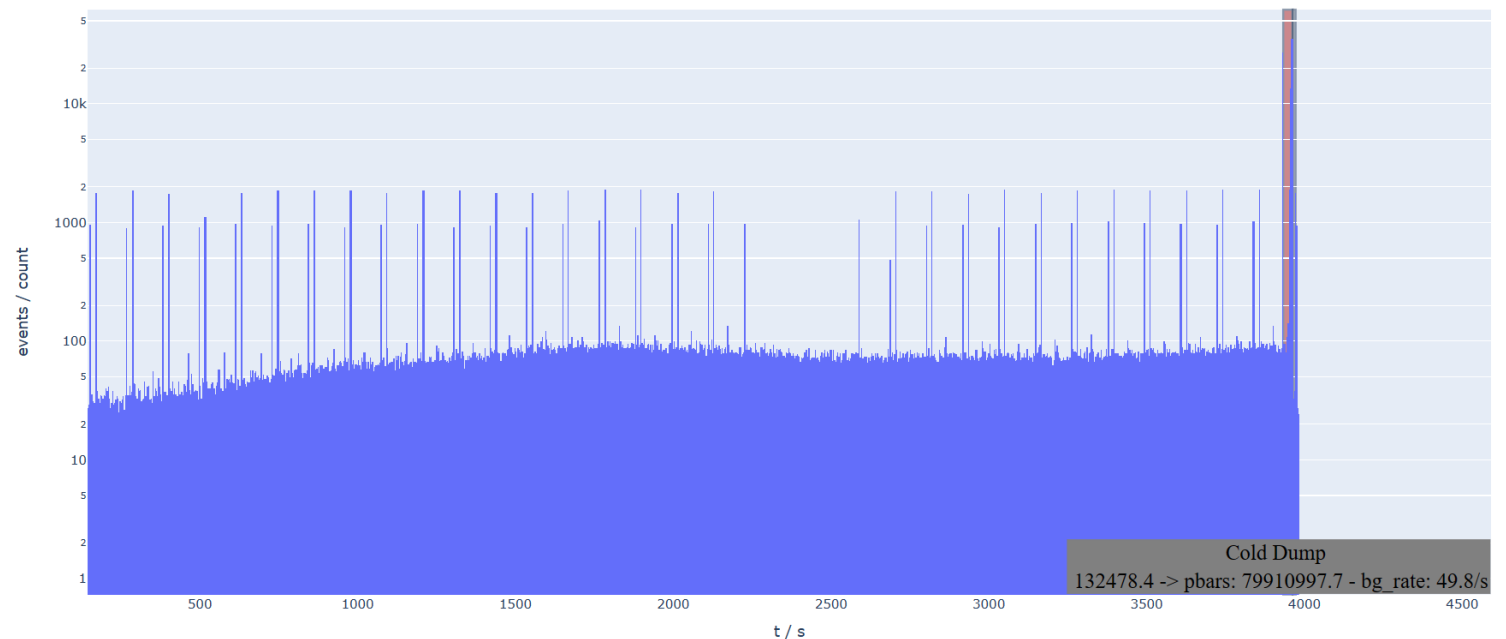
- Article -

# Efficient Accumulation of Antiprotons

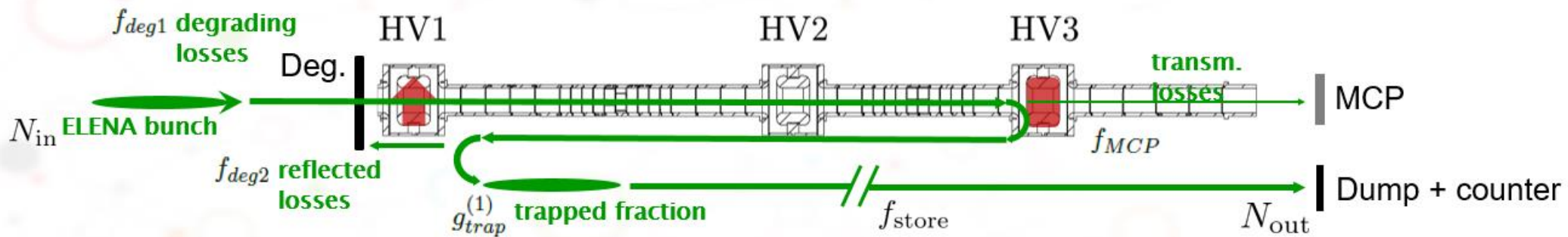
- Article -



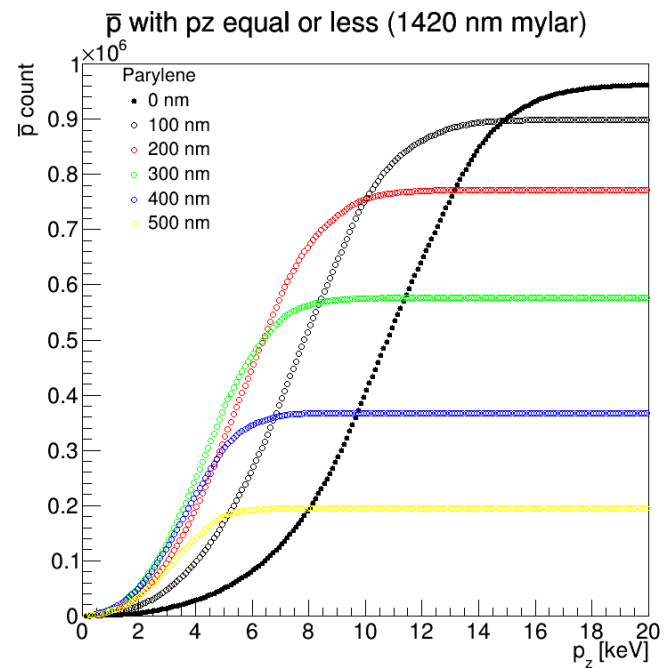
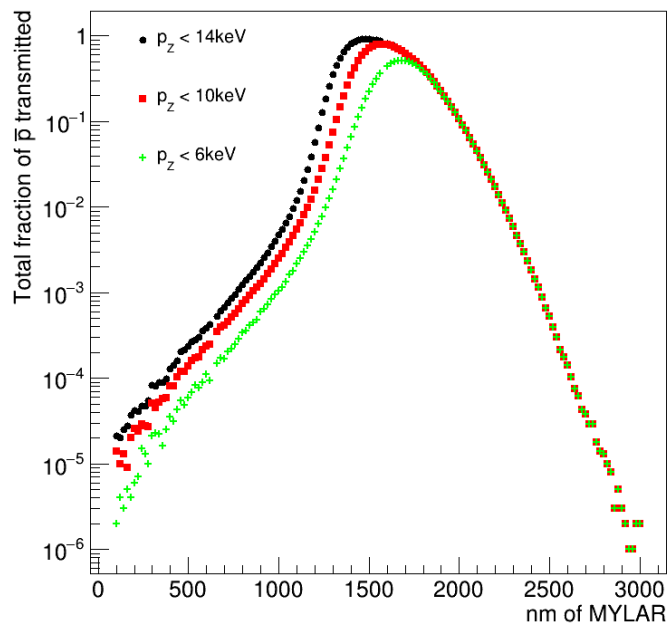
event clock SC1112, Run: 405247, Backgr. rate: 49.78

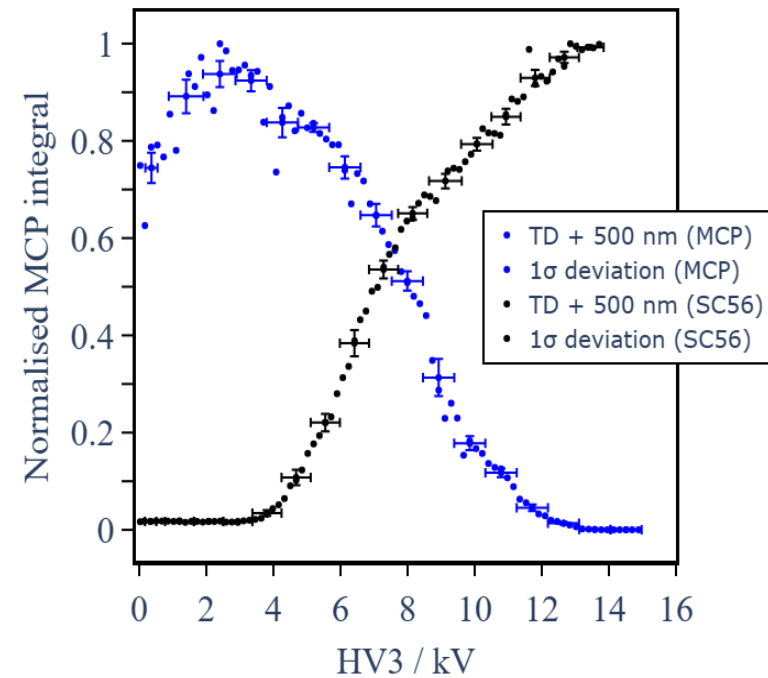
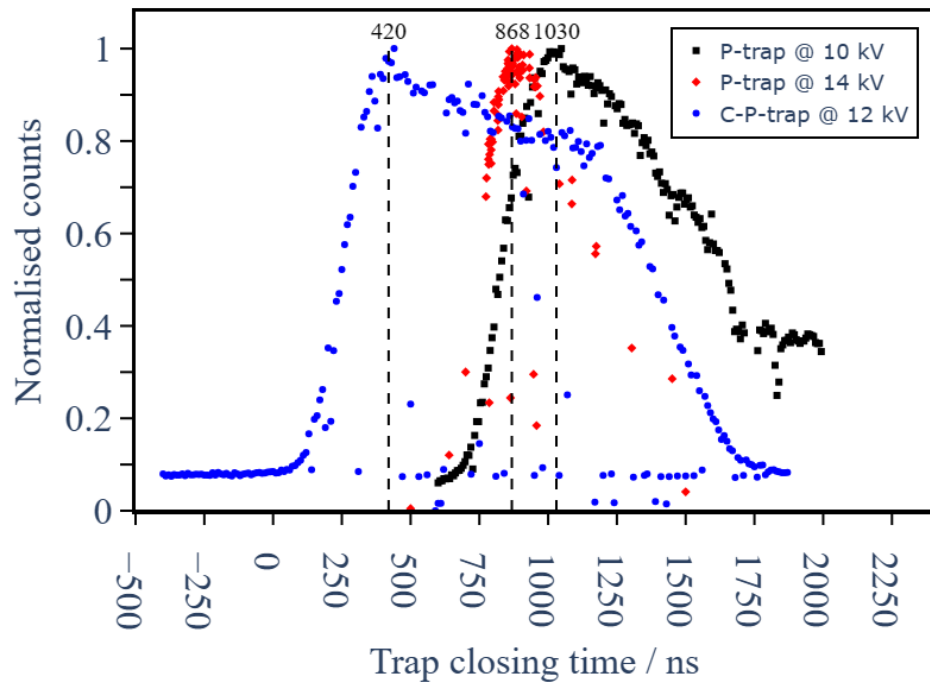
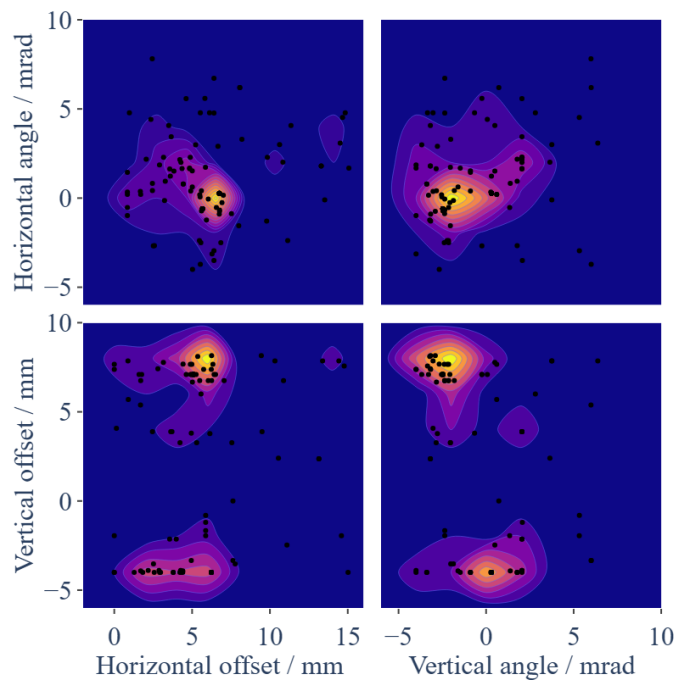
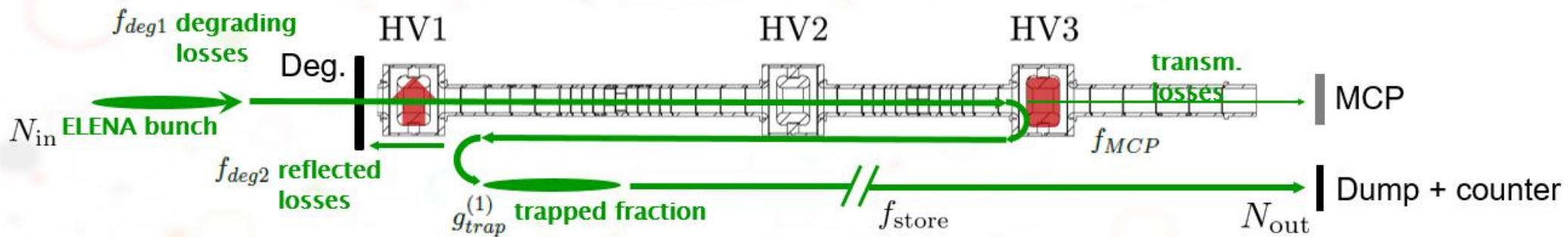


1. Catch
2. Hot Dump
3. Cool & Compress with RW
4. Accumulate in few eV trap



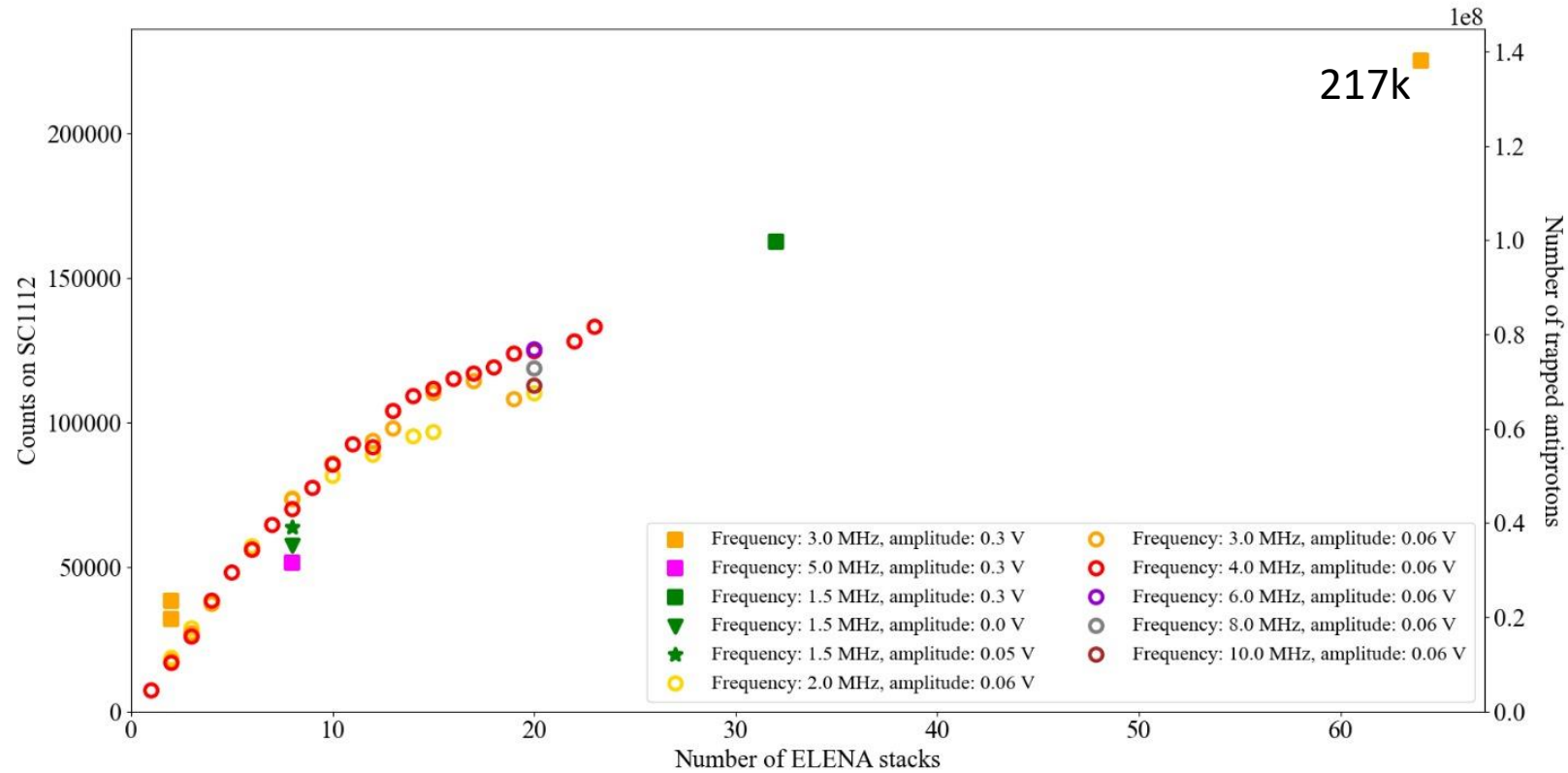
Trap efficiency





# Efficient Antiproton Trapping

- Article -



SC1112  $\rightarrow$  SC12: x ???

SC12  $\rightarrow$  #pbars: x 16.0

$\rightarrow$  Currently investigating!

# Efficient Antiproton Trapping

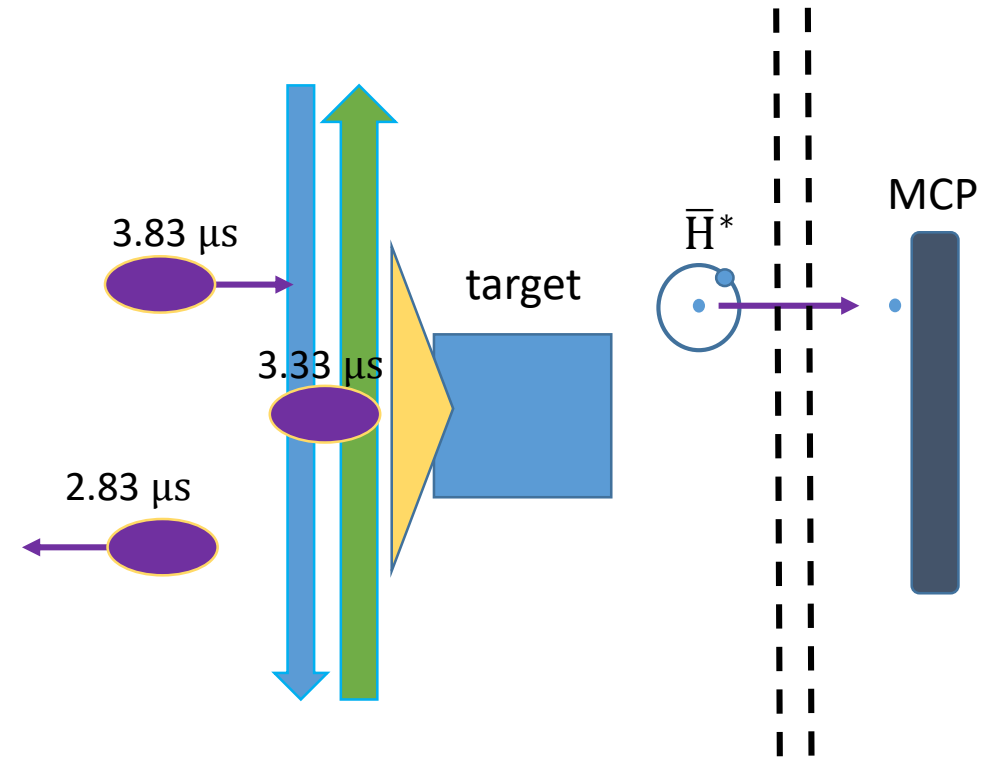
- Article -

**Listen to a proper presentation by Saiva  
in the next Monday meeting!!!**

# N x 4 measurement scheme of Hbar production

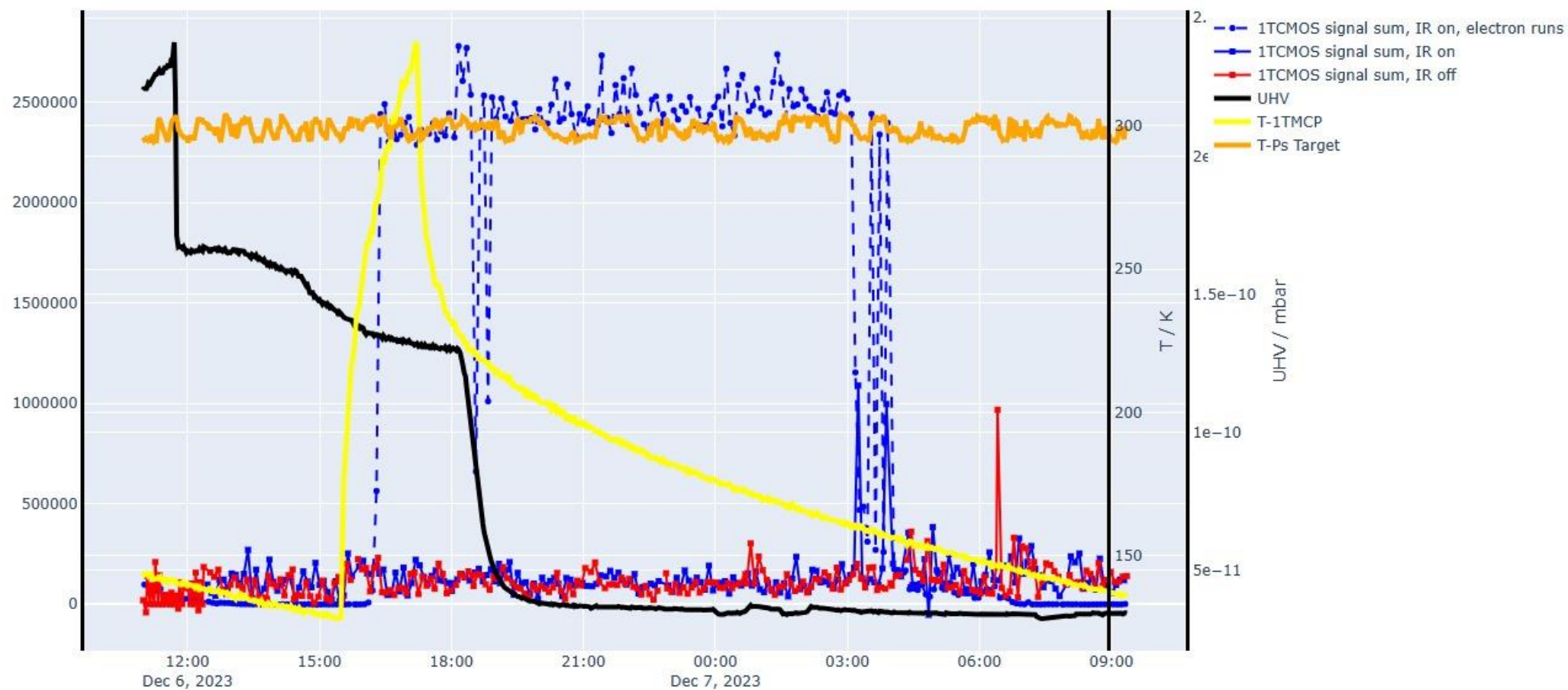
N x 4 scheme

IR	Positron-Pbar-Delay / $\mu\text{s}$
off	3.33
on	2.83
on	3.33
on	3.83



# 1TMCP “clogging like effect”

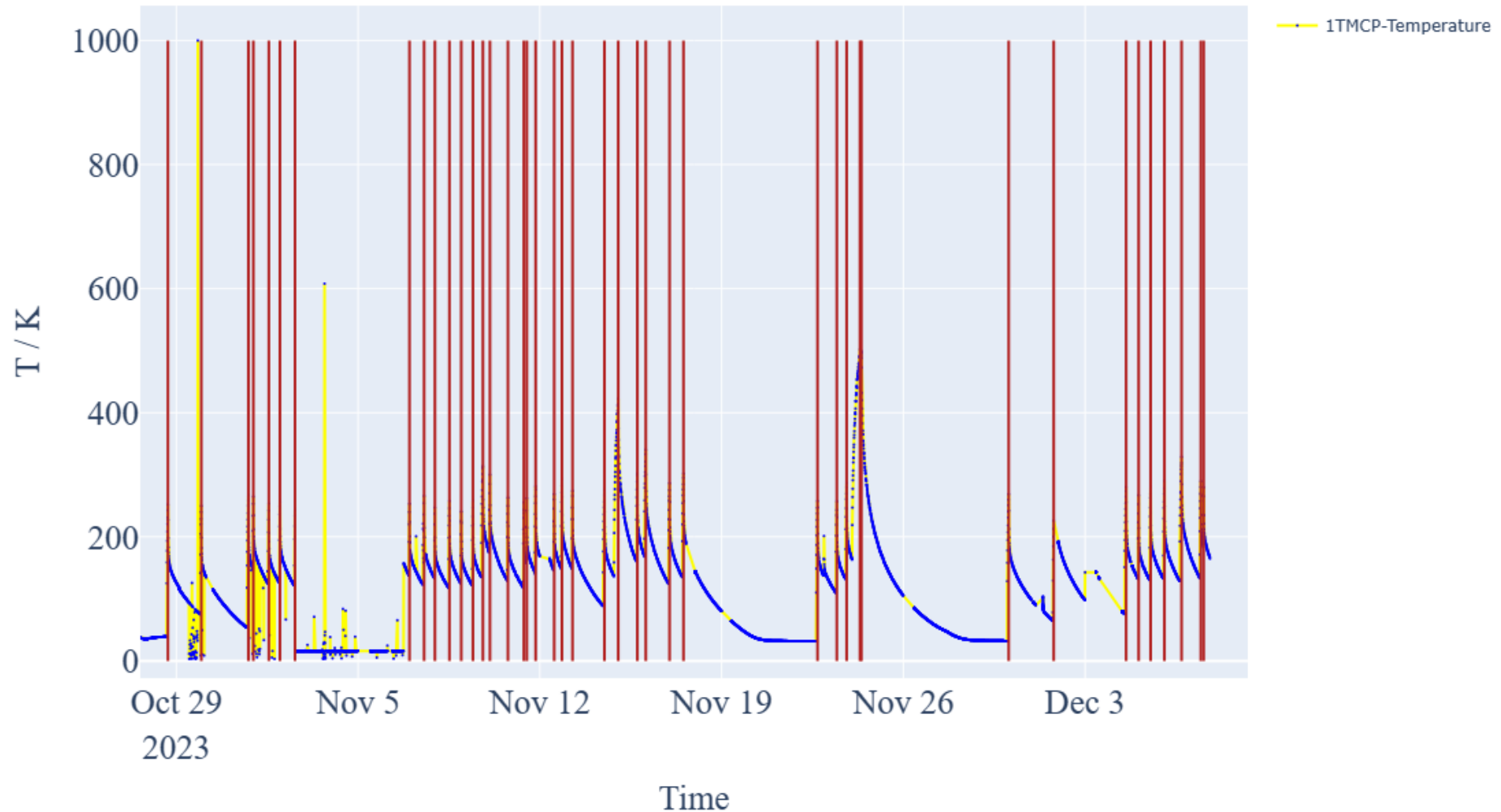
MCP clogging - 410424\_411997\_HL\_4th run





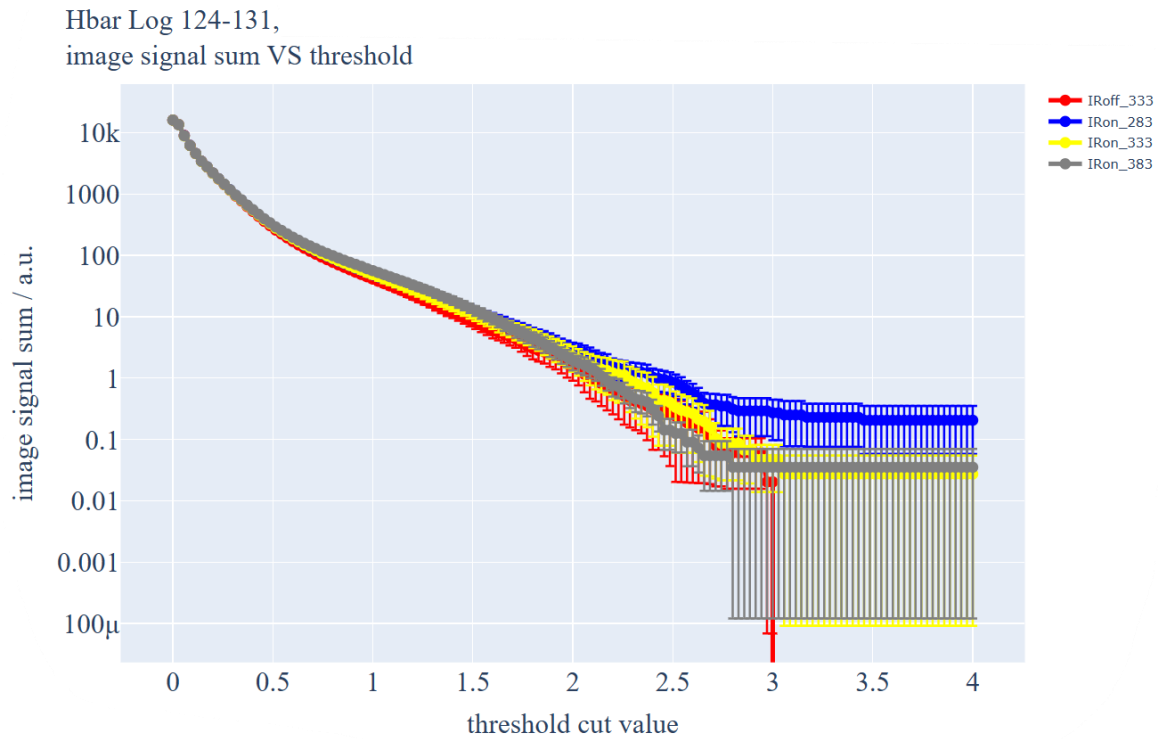
# Peak finding of T-1TMCP

1TMCP Temperature over run time

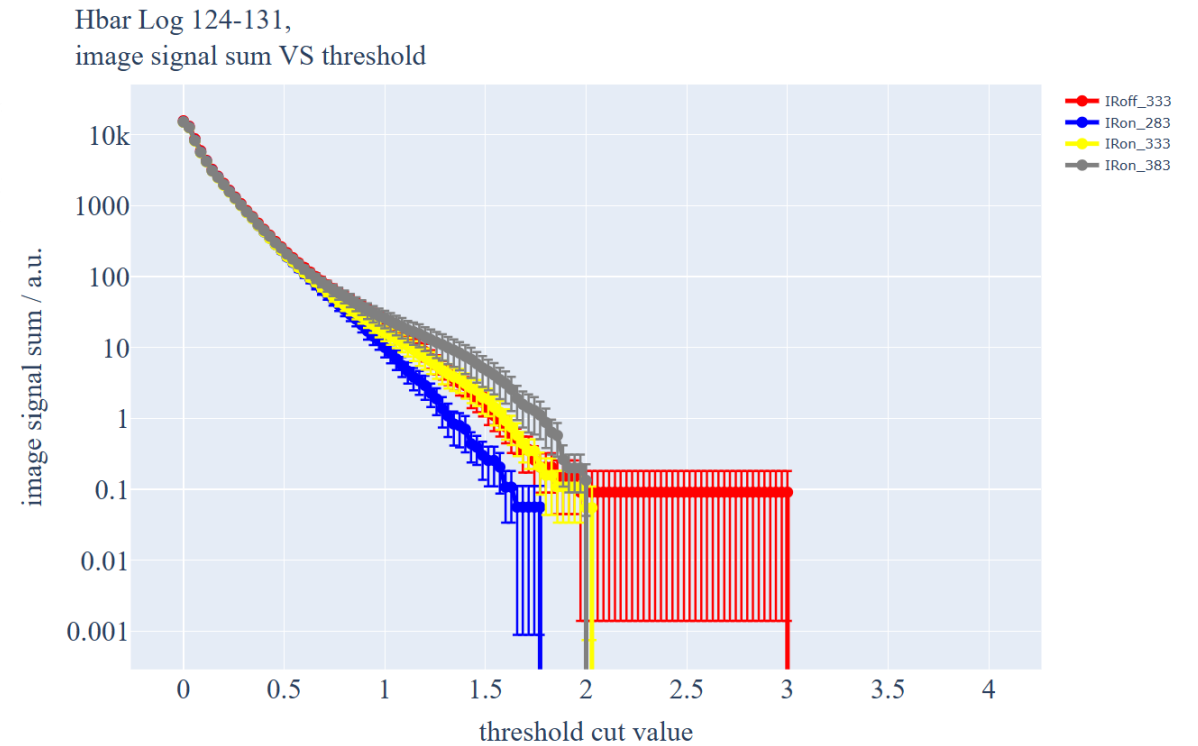


# $\bar{H}$ formation

Nx4 measurement scheme,  
All 600 runs



Inside golden window, 150 runs



ALPACA



-

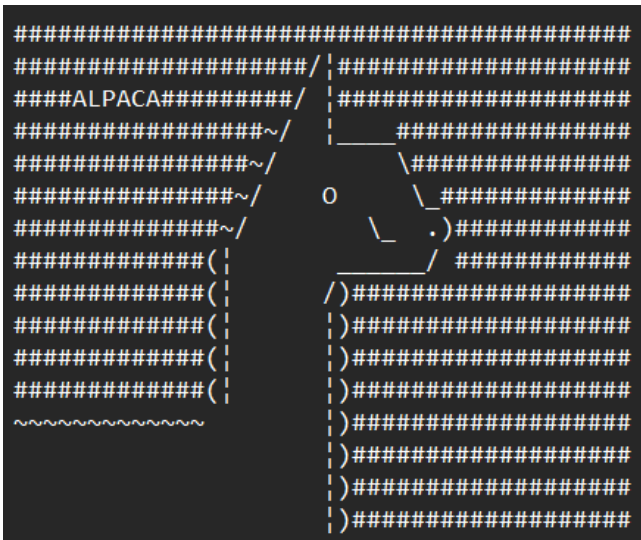
# Status & Future

-

Status April 2024

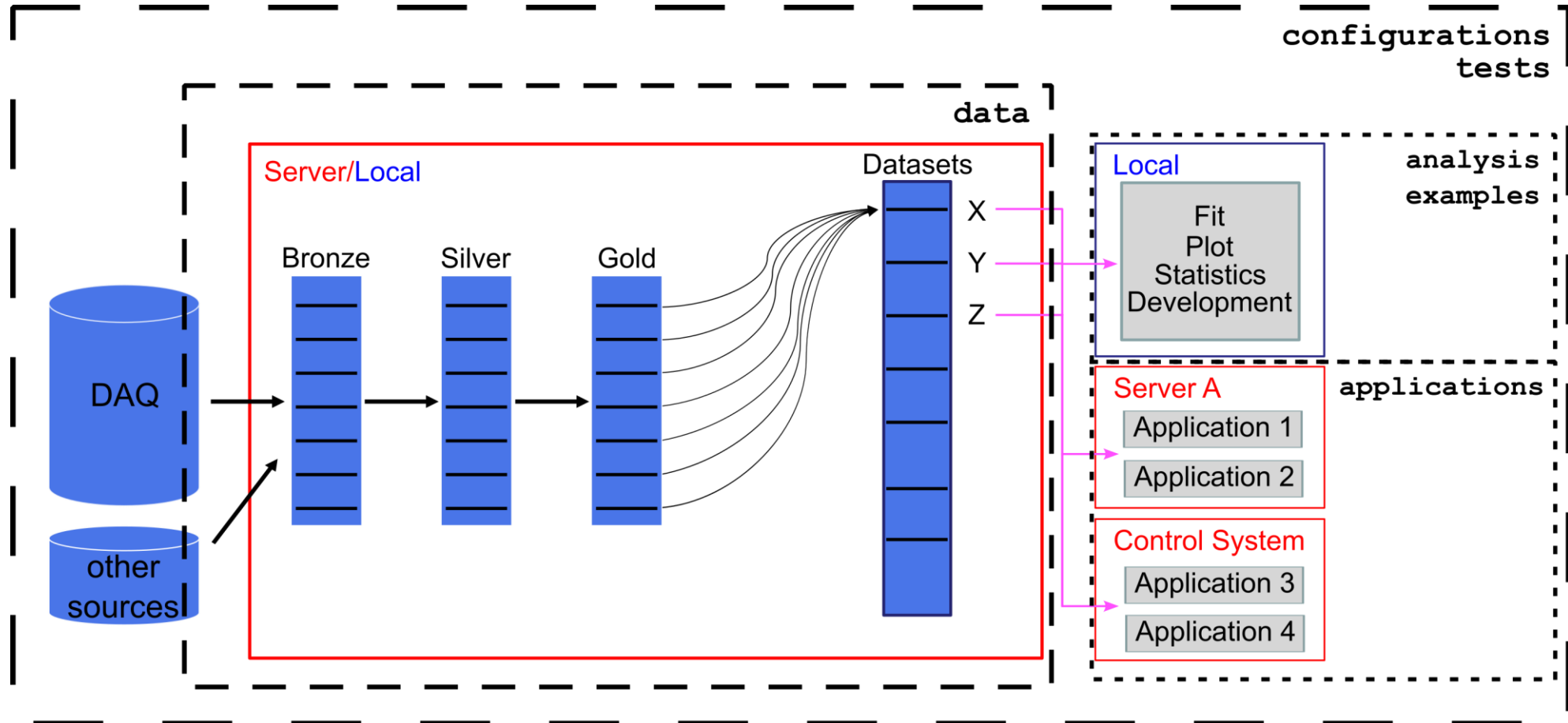
-

Tassilo Rauschendorfer



# ALPACA – Architecture

(Scipy, Numpy, Plotly)



# Documentation

ALPACA API documentation

aegis.docs.cern.ch/ALPACA/

## Index

- ALPACA - All Python Analyses Code of Aegis
  - Project Overview
    - Architecture
      - Getting started
        - Linux Systems
          - install python and git
          - close and install the project
        - Windows Systems
          - install python and git
          - close and install the project
      - Examples
      - Users & Developers
        - General Guidelines for developers
        - Test, Build and Deploy
      - Deployment Setup (Maintainers only) on aegisonline
      - Features
      - Contributing
      - Authors and acknowledgment
      - Project status

## Package ALPACA

### ALPACA - All Python Analyses Code of Aegis

Read this README along with the full code documentation at: <https://aegis.docs.cern.ch/ALPACA>  
Read the FAQ for Users at: [https://aegis.docs.cern.ch/ALPACA\\_FAQ/users](https://aegis.docs.cern.ch/ALPACA_FAQ/users)  
Read the FAQ for Developers at: [https://aegis.docs.cern.ch/ALPACA\\_FAQ/developers](https://aegis.docs.cern.ch/ALPACA_FAQ/developers)

### Project Overview

ALI Python Analyses Code of Aegis (ALPACA) is a library written in Python for the Aegis experiment at CERNs Antiproton Decelerator (AD). The Library provides a pipeline architecture to process data of the experimental runs from an initial raw to a final gold state. A set of high level functions is provided for easing direct analysis on user defined datasets and execute scripted applications.

### Architecture

The following Figure depicts the data flow through the ALPACA framework.

```
graph LR
    DAQ[DAQ] --> Bronze[Bronze]
    OtherData[other data] --> Bronze
    Bronze --> Silver[Silver]
    Silver --> Gold[Gold]
    Gold --> Datasets[X, Y, Z]
    Datasets --> Local[Local: Filter, Fit, Plot, Statistics, Development]
    Datasets --> AegisOnline[AegisOnline: Continuous Analyses]
    Datasets --> Monkey[Monkey: Optimizer, Analyses]
```

ALPACA.data.pipelines.gold.CM

aegis.docs.cern.ch/ALPACA/data/pipelines/gold/CMOSDataAnalysis.html#ALPACA.data.pipelines.gold.CMOSDataAnalysis.CMOSDataAnalysis.get\_radial\_profile

## Index

### Super-module

- ALPACA.data.pipelines.gold

### Classes

- CMOSDataAnalysis
  - analyse
  - analyse\_background
  - get\_background\_mean\_std
  - get\_background\_normalised\_img
  - get\_img\_observables
  - get\_line\_segment\_means
  - get\_radial\_profile
  - get\_roi
  - get\_roi\_observables
  - make\_sub\_array

```
def get_radial_profile(self, data: dict) -> dict
```

EXPAND SOURCE CODE

Determines the center pixels of the signal on the CMOS. From the center the algorithm runs rings with increasing radius but constant width over the image and determines observables for each ring. This way one can analyse the radial profile originating from the radial geometry of the trap.

```
def get_radial_profile(self, data: dict) -> dict:
    """
    Determines the center pixels of the signal on the CMOS. From the center the algorithm runs rings wit
    increasing radius but constant width over the image and determines observables for each ring. This w
    can analyse the radial profile originating from the radial geometry of the trap.
    """
    # get xhist and yhist
    xhist = data['background_corrected']['xhist']
    yhist = data['background_corrected']['yhist']

    # get the background corrected image
    array = data['background_corrected']['background_normalised_img']

    # get threshold from deviation by 4 sigma
    threshold = data['background_corrected']['std_background'] * 4 / data['background_corrected']['mean_

    # make binary array
    binary_array = array > threshold

    # get sums of pixels with 1
    sum_y, sum_x = np.indices(array.shape)

    # get weighted sums
    weighted_sum_y = (binary_array * sum_y).sum()
    weighted_sum_x = (binary_array * sum_x).sum()

    # get total number of pixels with 1
    number_pixels = binary_array.sum()
```

```
def get_roi(self, data: dict)
```

EXPAND SOURCE CODE

Determines the roi (region of interest) using edge recognition.

```
def get_roi_observables(self, data: dict)
```

EXPAND SOURCE CODE

# Continuous Integration – Continuous Deployment

AEgIS > Python analyses > Pipelines

All 851 Finished Branches Tags

Clear runner caches CI lint Run pipeline

Filter pipelines  Show Pipeline ID ▾

Status	Pipeline	Created by	Stages	
<span>✖ Failed</span> ⌚ 00:01:05 📅 4 months ago	Added the masking also to the other Coolin... #6127786  benji		<span>✖</span> <span>»</span>	
<span>✖ Failed</span> ⌚ 00:01:43 📅 4 months ago	Added the masking feature #6127669  benji		<span>✖</span> <span>»</span>	
<span>✔ Passed</span> ⌚ 00:12:02 📅 4 months ago	Merge branch 'develop' into 'main' #6126992  main		<span>✔</span> <span>✔</span> <span>✔</span>	
<span>✔ Passed</span> ⌚ 00:07:21 📅 4 months ago	Merge branch '107-deploy-dashboards' into... #6126927  develop		<span>✔</span> <span>✔</span> <span>✔</span>	
<span>✔ Passed</span> ⌚ 00:12:19 📅 4 months ago	clean and add documentation of gitl... #6126651  107-deploy-dashboards latest		<span>✔</span> <span>✔</span>	

Stage: build

- ✔ build\_library
- ✔ pages

Stage: deploy

- ✔ deploy\_alpaca\_das...
- ✔ deploy\_bayesian\_o...

Stage: test

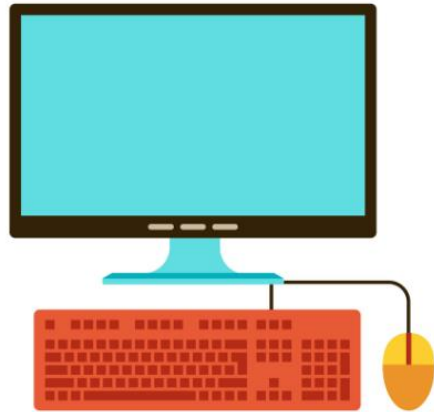
- ✔ run\_tests
- ✔ test\_CL\_tools

Stage: deploy

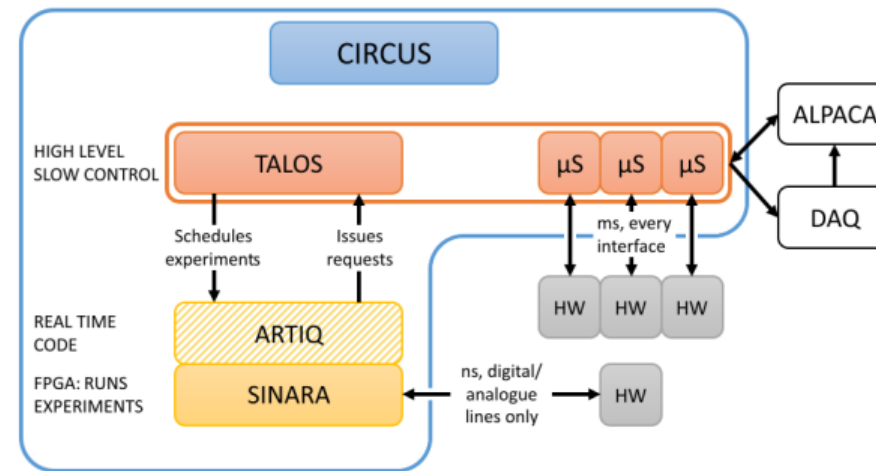
- ✔ pages:deploy

# Where does the ALPACA currently live?

Personal Computers



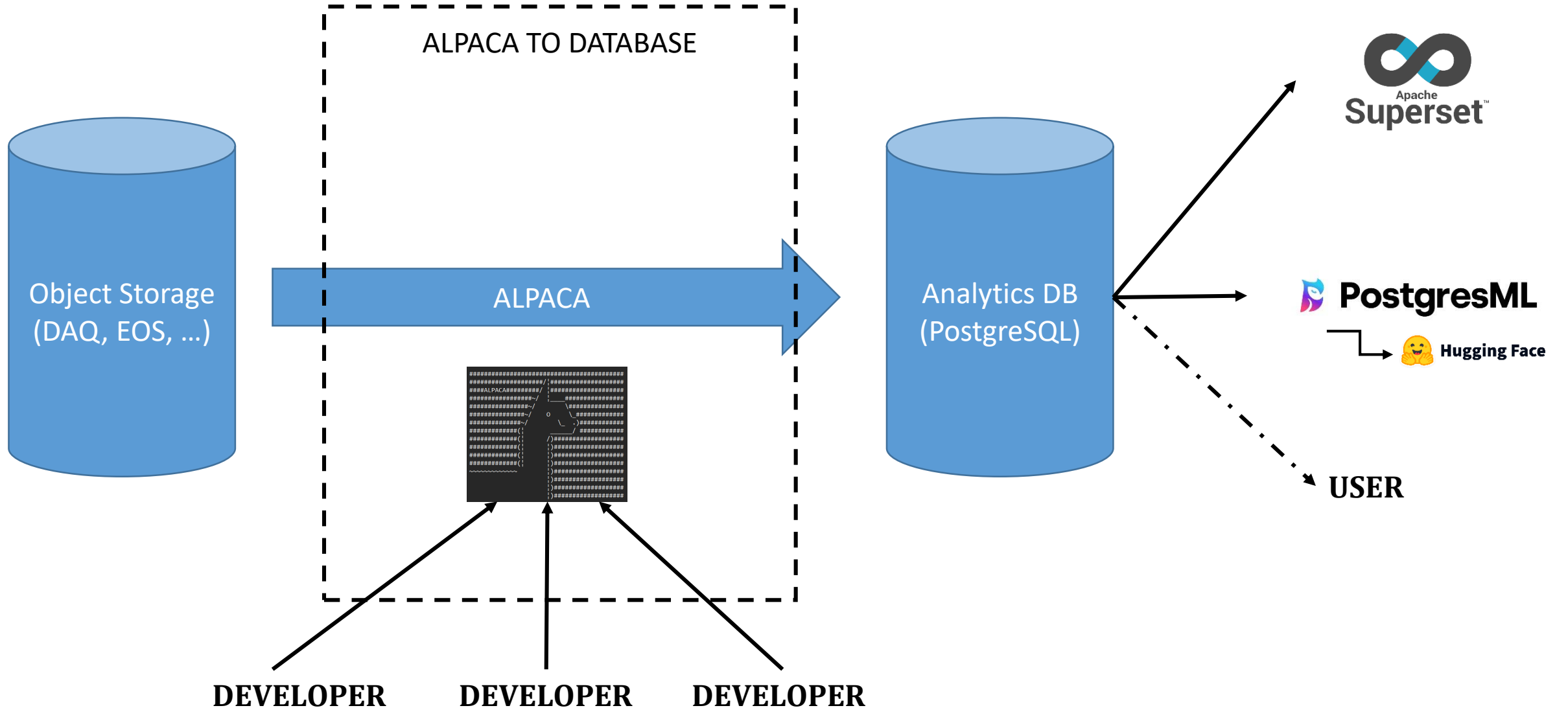
Aegis Control System



aegisonline

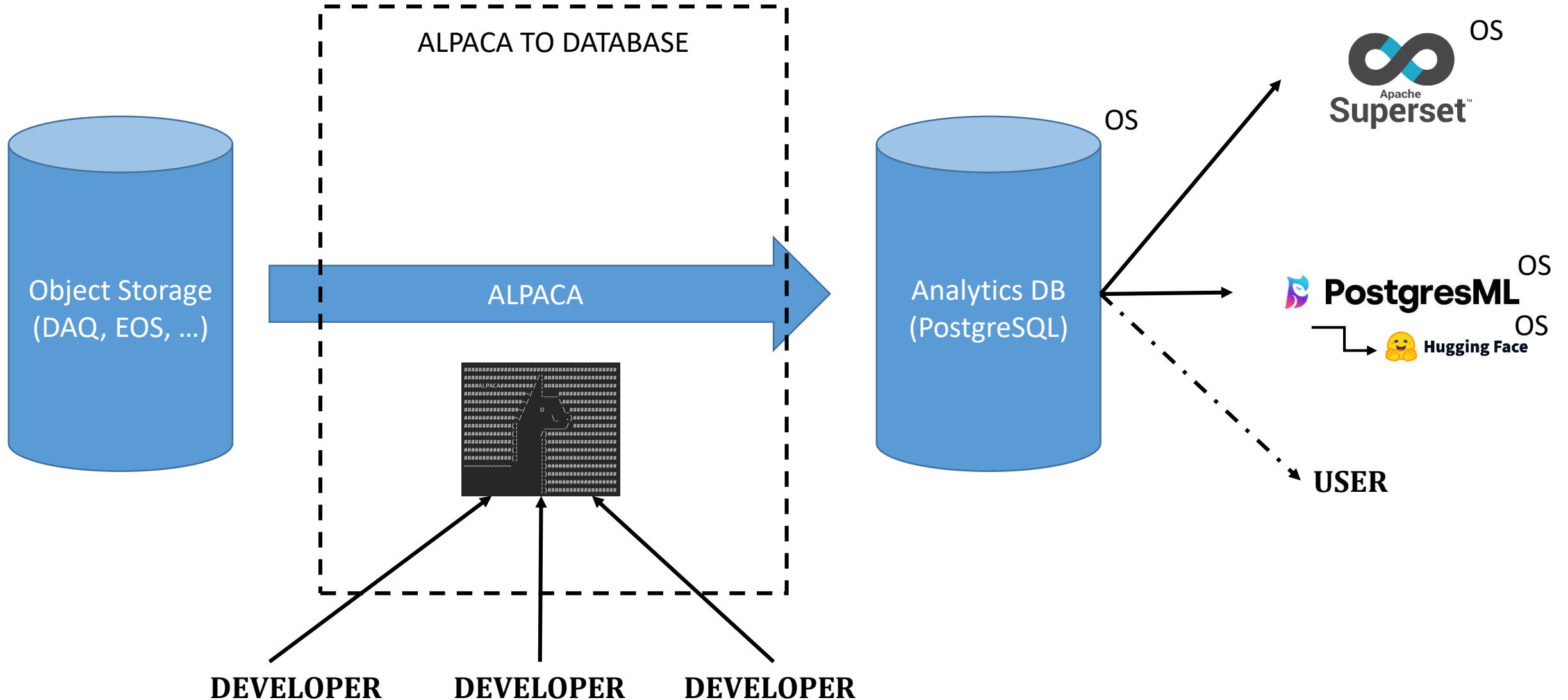


# Full Stack Integration





# Full Stack Integration - Powered by Open Source -



# Alpaca to Database

(web app)

The screenshot shows a web browser window with the title "ALPACA TO DATABASE". The browser tabs include "Superset", "AEGIS DAQ control", "RunLog - Missing Catch D", and "ALPACA TO DATABASE". The address bar shows "127.0.0.1:5003". The page content is divided into three main sections:

- Continuous Analyses:** A control panel with a green "OK" indicator and a "Stop" button. The log shows: "2024-05-06 17:39: Started Continuous Analyses", "2024-05-06 17:39: Latest run in DAQ: 414514. Latest run in PostgresDB: 414481.", "2024-05-06 17:39: Latest run in DAQ: 414514. Latest run in PostgresDB: 414482.", "2024-05-06 17:40: Stopped Continuous Analyses", "2024-05-06 17:40: Process with PID: 1577428 successfully terminated.", "2024-05-06 18:17: Started Continuous Analyses".
- Rewrite DB from DAQ:** A control panel with a red "Stopped" indicator, "Update ALPACA" and "Start" buttons. The log is empty.
- Custom Run Analyses:** A control panel with a red "Stopped" indicator, input fields containing "409524", and a "Start" button. The log shows: "2024-05-06 17:36: Starting analyses of specified runs", "2024-05-06 17:36: finished well.", "2024-05-06 17:37: Starting analyses of specified runs".

Each section includes a terminal window with log output, a status indicator (green or red), and a database icon. A purple sidebar on the right contains a "DB Status" section with the following information:

- Database connected: yes
- Total: 438.58, Used: 103.24, Free: 312.99 (GB)
- Number of Runs stored: 2461
- last Run in DAQ: 414515
- first Run in DAQ: 334449
- last Run in DB: 414482
- first Run in DB: 373007
- last modified Run in DB: 414482 @2024-05-06 17:39:49.996330
- dt of last 2 modified runs in db
- dt of last 2 runs in DAQ

# Observables over many Runs Draft EDIT DASHBOARD

Usual Observables All about Positrons All about Pbars All about HCIs

### Pbar Settings

Show 200 entries Search 2461 records...

run_number	run_dir_creation	Batman_O_NegHV_Ch1	Batman_O_NegHV_Ch2	Batman_O_Elena_V_Angle	Batman_O_Elena_H_Angle	Batman_O_Elena_V_Offset	Batman_O_Elena_H_Offset	Batman_O_PbarCoolingTime
414482	2024-05-05 19:43 — 2024-05-05 19:44	N/A	N/A	N/A	N/A	N/A	N/A	N/A
414481	2024-05-05 19:26 — 2024-05-05 19:27	N/A	N/A	N/A	N/A	N/A	N/A	N/A
414480	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

1 2 3 4 5 6 7 ... 13



### General Settings

Show 200 entries Search 2461 records...

run_number	run_dir_creation	Batman_O_DAQ	beam_Intensity	H_angle
414482	2024-05-05 19:43:53	N/A	4896400	
414481	2024-05-05 19:26:59	N/A	4896400	

1 2 3 4 5 6 7 ... 13

# Specific Run ☆ Draft

[EDIT DASHBOARD](#)

Run Number

411999

Started run at

2023-12-07 09:24:28

Beam Intensity from elena

## No data

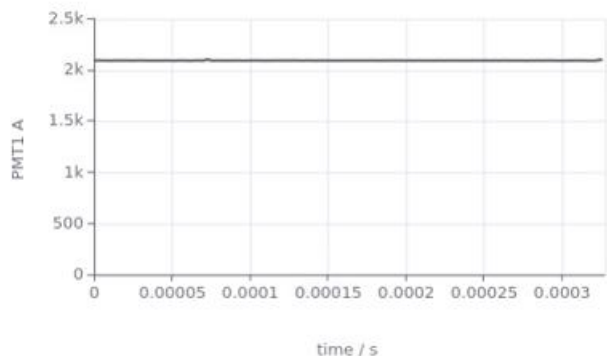
No data after filtering or data is NULL for the latest time record

### Metadata

MAX(Metadata_Intensity_ekspla)	MAX(Metadata_detuning_ekspla)	MAX(Metadata_delay_ekspla)	MAX(Metadata_Class)	MAX(Metadata_RealNoP)
N/A	N/A	N/A	N/A	N/A

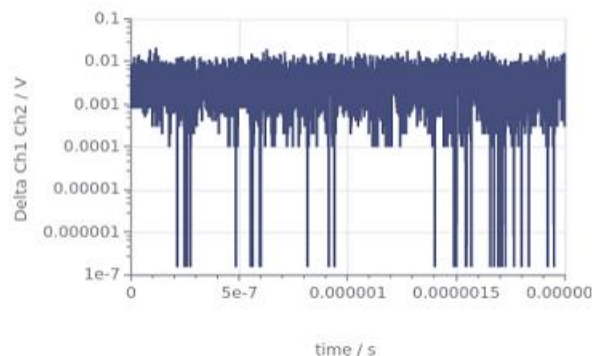
### PMT1

MAX(PMTDigi1\_0\_A) All Inv



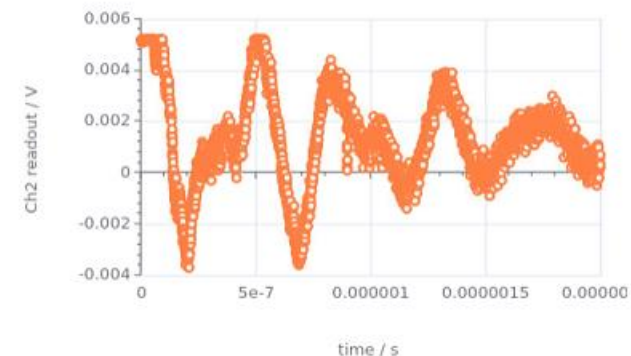
### Captorius3\_Ch2\_deltaV

MAX(Captorius3\_0\_Ch1\_2\_delta\_V) All Inv



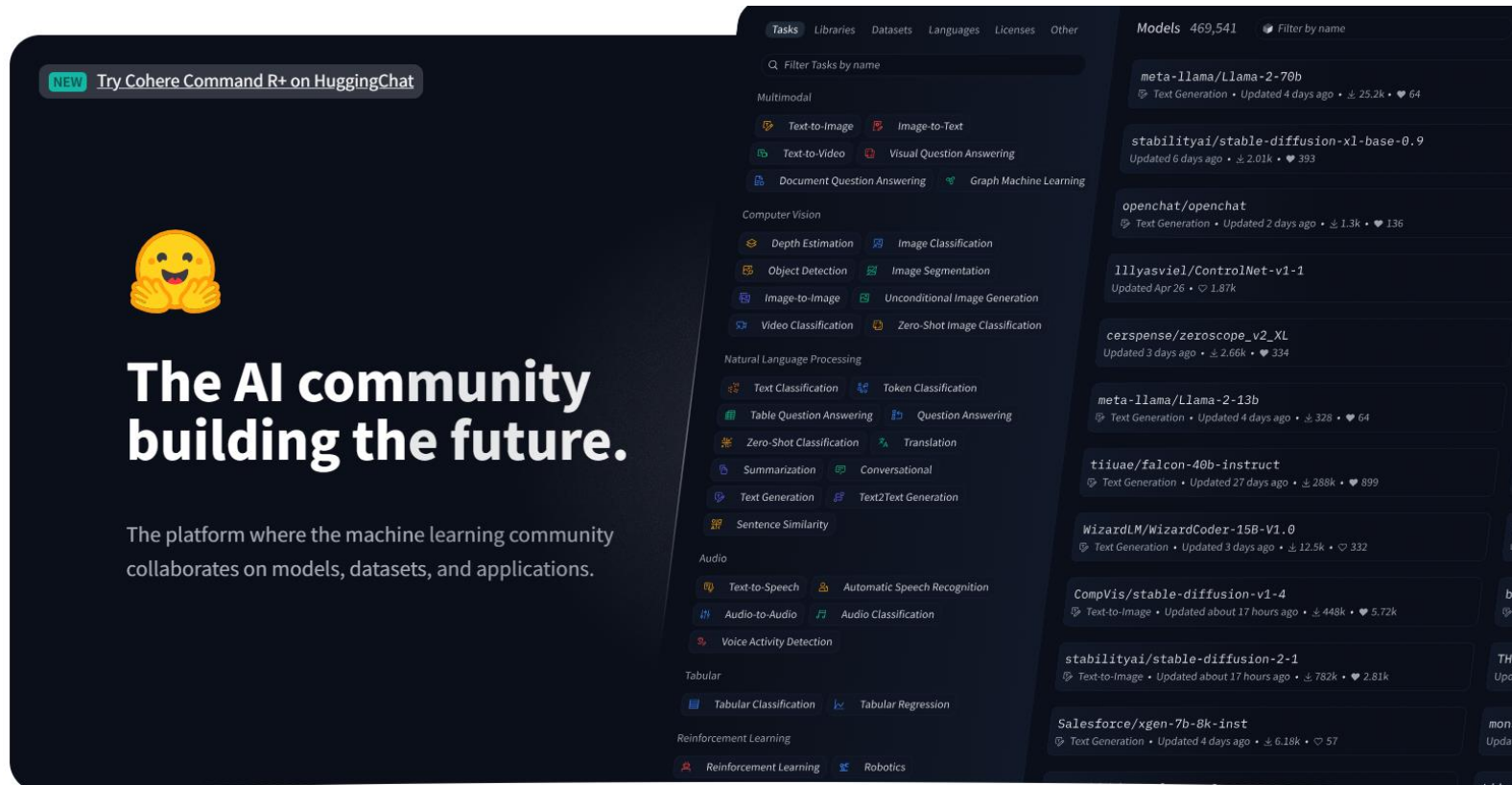
### Captorius3\_Ch2\_readout

MAX(Captorius3\_0\_Ch2\_V) All Inv



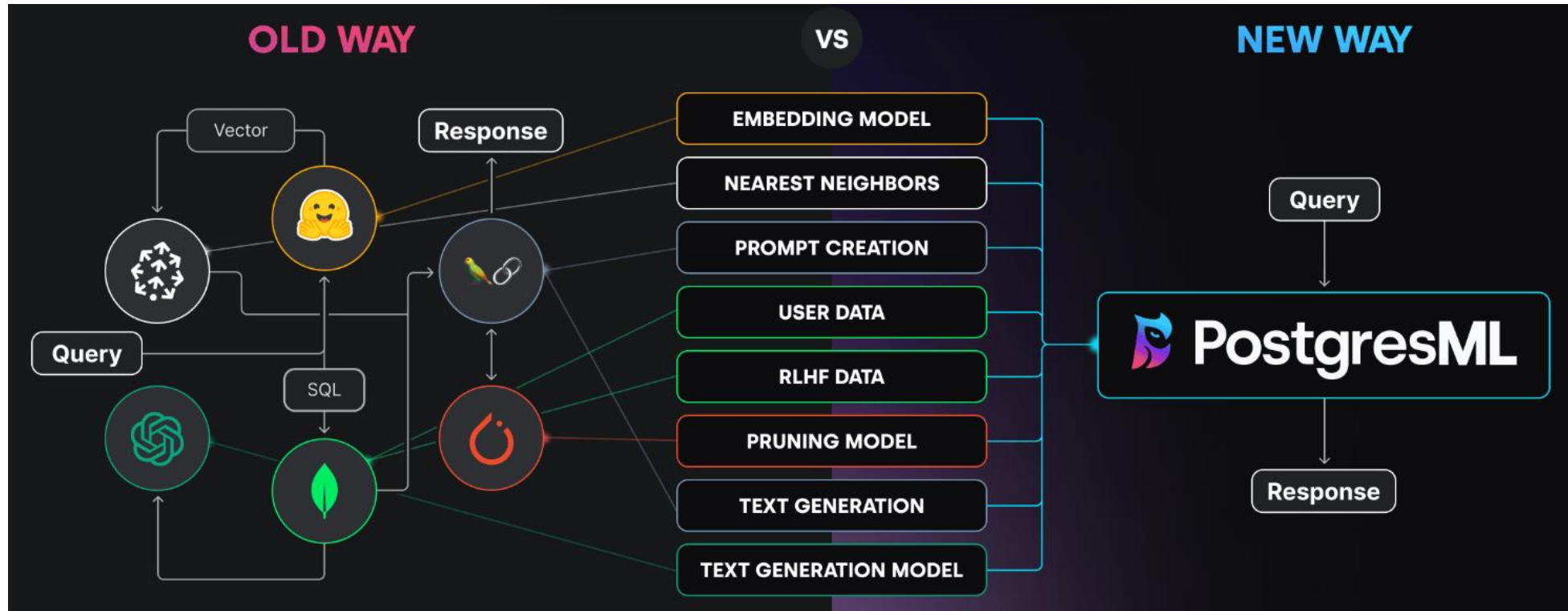


- A **no-code interface** for building charts quickly
- A powerful, web-based **SQL Editor** for advanced querying
- A **lightweight semantic layer**
- **Support for nearly any SQL** database or data engine
- wide array of **beautiful visualizations**
- **security roles and authentication**
- **API**



- ML: More parameters → Better performance // Breakthrough of GPTs (LLNs)
- High economic pressure:
  - OpenAI (CS) → GPT4: 1.7T parameters 🤗
  - Meta & Google (OS) → Llama3: 8-70B parameters
  - Models bigger than data... → Bring them to the DB!

# PostgresML



- PostgresML
  - Import data / Export models
  - Notebooks: ML with SQL
  - Full integration of 🤗 Hugging Face
  - Training
  - Inference

# SQL Extension

The SQL extension provides end-to-end ML & AI functionality from inference to deployment. It can be used in any combination to implement bespoke models across use cases.

## AI

### **pgml.embed()**

Generate high quality embeddings with faster end-to-end vector operations without an additional vector database.

### **pgml.transform()**

Perform dozens of state-of-the-art natural language processing (NLP) tasks with thousands of models. Serve with the same Postgres infrastructure.

### **pgml.tune()**

Fine tune open-source models on your own data.

## ML

### **pgml.deploy()**

Release trained models when ML quality metrics computed during training improve. Track model deployments over time and rollback if needed.

### **pgml.predict()**

Batch predict from data in a table. Online predict with parameters passed in a query. Automatically reuse pre-processing steps from training.

### **pgml.train()**

Pre-process and pull data to train a model using any of 50 different ML algorithms.



# Via Notebooks

```
1 1 SELECT pgml.transform(  
2     'translation_en_to_fr',  
3     inputs => ARRAY(  
4         'Welcome to PostgresML!',  
5         'The next generation ML Database'  
6     )  
7 ) AS french;
```

**french**

```
[{"translation_text": "Bienvenue à PostgresML!"}, {"translation_text": "Base de données sur le ML de la prochaine génération"}]
```

3.057067s

# Why interesting?

- Unique **Control System**
  - Code driven
  - Autonomous
  - Self-driven (closed feedback loops)
- Our Physics data
  - Sequential & iid
  - images, spectra, annihilation events
  - Raw & Analysed
- Research on ML in Physics of **high interest**
- specialised niche to study the intersection of ML/AI & low energy atomic physics, antimatter, plasma physics

# What's next?

## Analytics Object Storage:

- Possible to store processed images
- Gravity Measurement with 6 GB images!

## Superset:

- Opening up to outside CERN
- Access distribution (training)
- Upgrade to 4.0

## PostgresML:

- Run models in DB
- Integrate Models from Hugging Face
- Deploy models
- Experiment the use-cases:
  - Coding
  - Analyses
  - Model Training not possible on aegisonline  
→ More GPU power needed