

Titania - how to structure detector monitoring

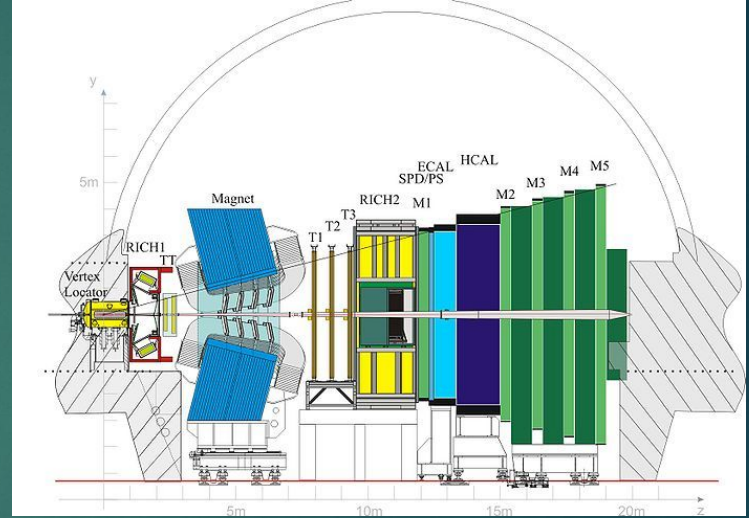
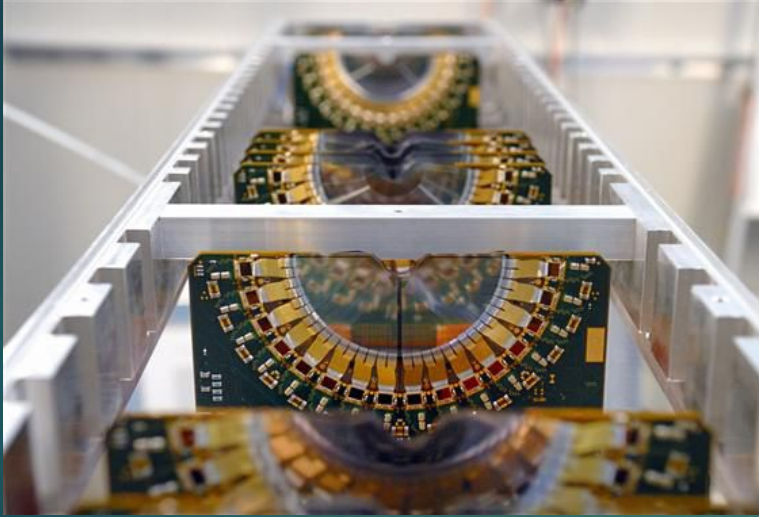
on behalf of LHCb

JAKUB KOWALSKI – AGH UST, LHCb-VELO
MACIEJ W. MAJEWSKI – AGH UST, LHCb-VELO



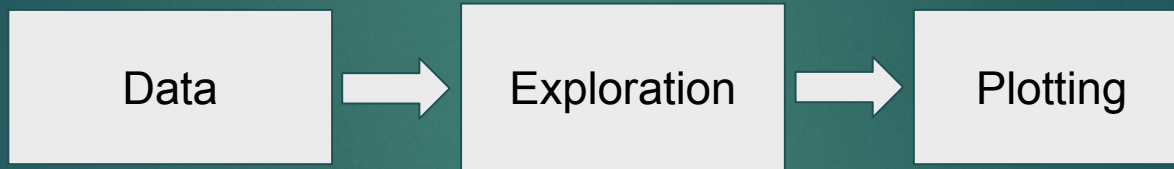
VELO

2

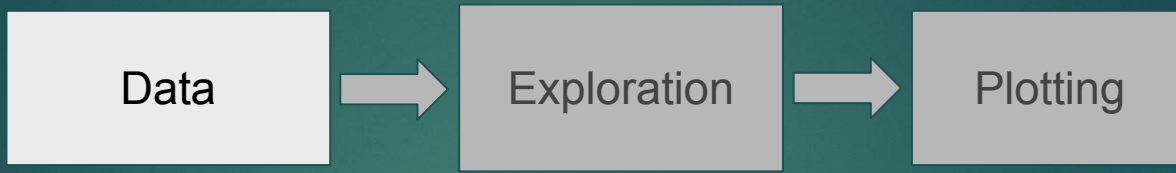


Monitoring, the code structure

3



Monitoring, the code structure

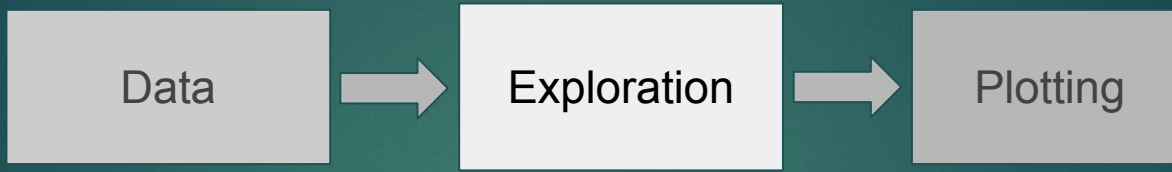


Data

- You need to either get the data from some kind of database
- Read the data into RAM
- You also probably need to prepare the data (to be able to display it or make it possible to browse it)

Monitoring, the code structure

5

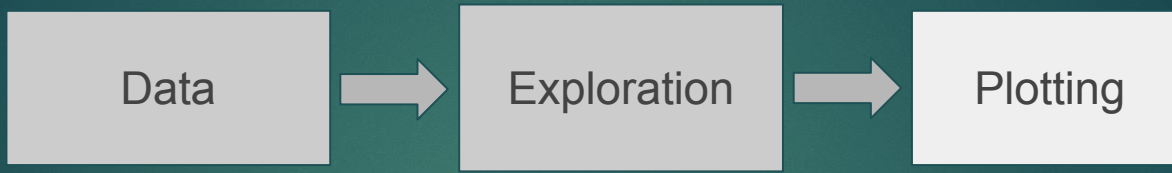


Exploration

- You want to be able to choose different ways of plotting data
- You want to be able to choose different data sources

Monitoring, the code structure

6



Plotting

- Making the actual graphs, charts, plots, histograms

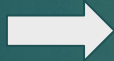
Monitoring, the code structure

7

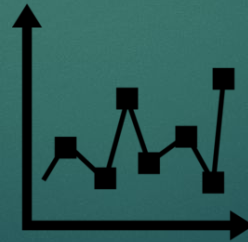
Data



Exploration

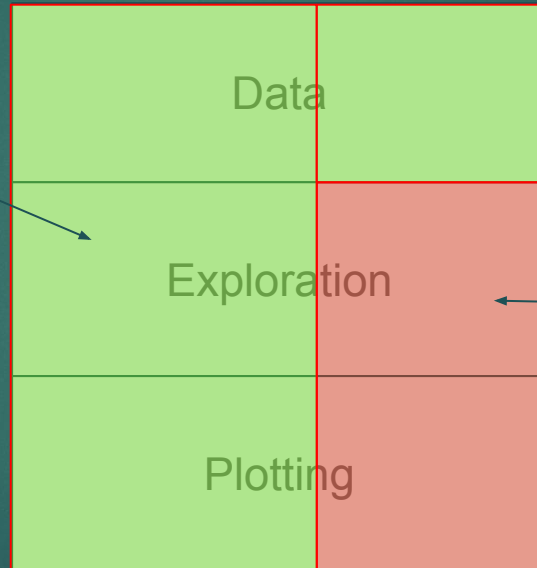


Plotting



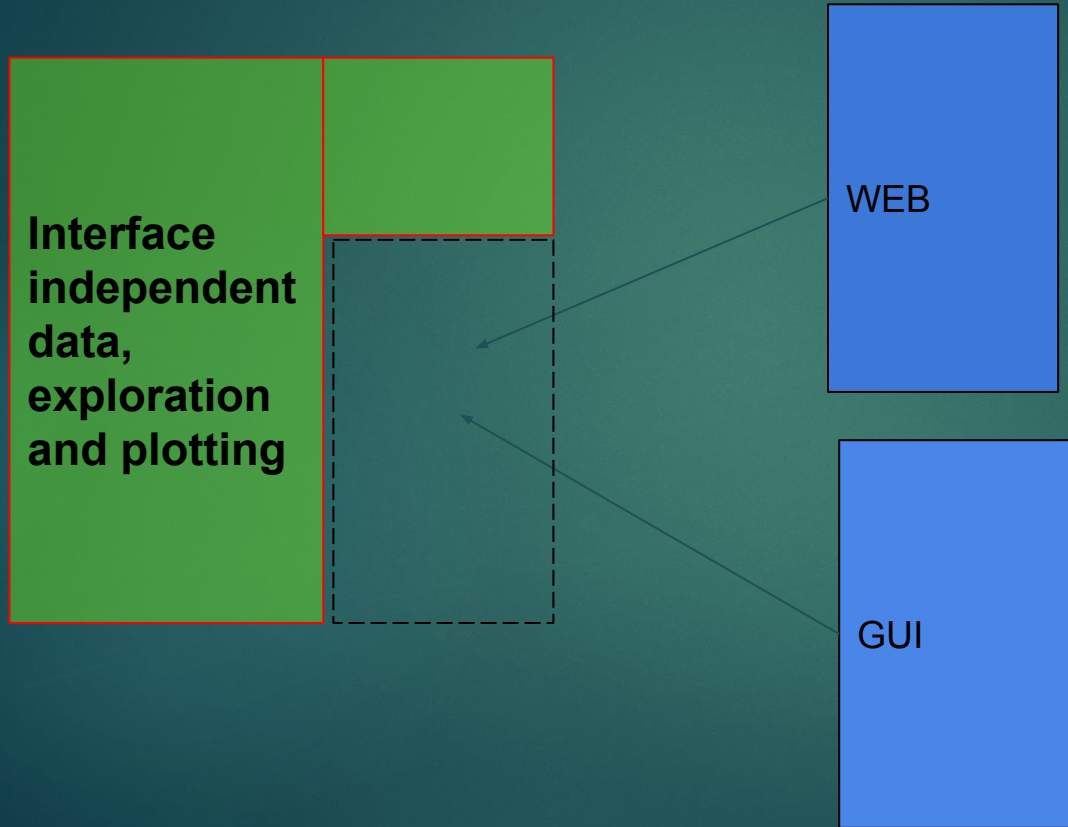
Monitoring, the code structure

Does not depend on graphical interface



Depends on graphical interface

Monitoring, the code structure



If we will write everything in well structured way, this will be easy to achieve.

Monitoring, the code structure

10

What i mean, by code being well structured;

- You don't need to write additional lines inside old code to add new code.

What i mean by reusable code;

- You don't need to write additional lines inside the code to use it for something new.

What i mean by readable code;

- You can tell exactly what is the code doing by the first look

TITANIA - motivation

- it would be good to have more advanced plotting capabilities (web interfaces have some limitations)
- we may need more specific stuff for special analyses
- TITANIA - is a proposed name for this project
- to be perfectly clear; as for now, it does not contain any root-histograms-reading related stuff, it does not contain any monitoring algorithms. It contains exemplary structure, some basic classes and functions.

Demo

Mind you, this is just an example, how you can use the underlying framework.

The most important piece of information is after the demo.

The actual important stuff

So, what it looks like is not important as much.

This is a generic piece of software, not the detector specific project.

Now we will take a look inside the project. Keep in mind that some things, especially names of classes are not final, and may subject to change.

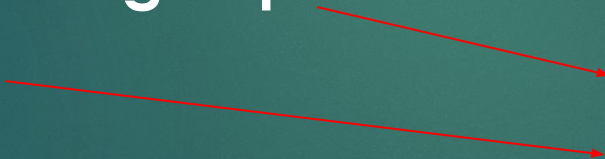
There are 3 main components to monitoring

- **Data:** This is what we see
- **Exploration:** An ability to choose between types of plots, changing the thing that we want to see, e.g. choosing correct module
- **Plotting:** plots graphs, charts and histograms

There are 3 components to monitoring

So we divided this project into core and implementation for each monitoring application:

- **monitoring implementation**
- **core**



VELO	fixed imports, added source
core	fixed imports, added source

There are 3 components to monitoring

Core is divided into 4 components

- **Data**
- **Exploration**
- **Plotting**
- **QtGui (for glueing it together)**

QtGui	fixed imports, added source
data	added installation option
panels	fixed imports, added source
plots	fixed imports, added source

DEMO

Advantages

18

Structure - Now you know where to put anything. If you need to add new database connection, you just add it to Data. If you need very specific kind of plot, you add it to plots. If you need to move around in specific way, you define new sidebar.

Reusability - If the interfaces are used, you can extend and reuse as many classes as you want, that are already implemented. You should use things that are already there.

Less dependencies

Anyone can contribute without making a mess

Can support multiple plotting frameworks

Can support interactive plots

Could support Monet in the future, or other waypoints

Conclusions

This piece of code **is not designed to view only one sensor** data
... but it will allow to do this right way, with minimal additional code

Main points:

- structured way of adding plots
- blocks of code, if properly developed are highly reusable
- anyone can add plots
- if you want to use it right away: <https://gitlab.cern.ch/mmajewsk/titania>
(soon moving to <https://gitlab.cern.ch/lhcb/titania>)
- we are more than happy to accept any suggestions or contributions at github issues

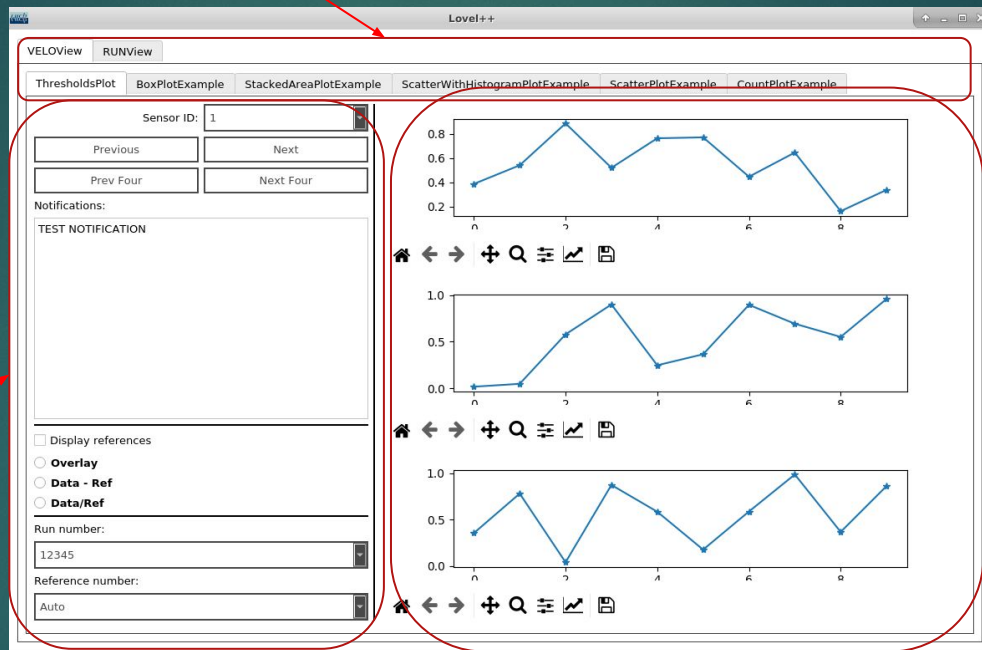
The end

Demo

Tabs

Sidebar

Plots

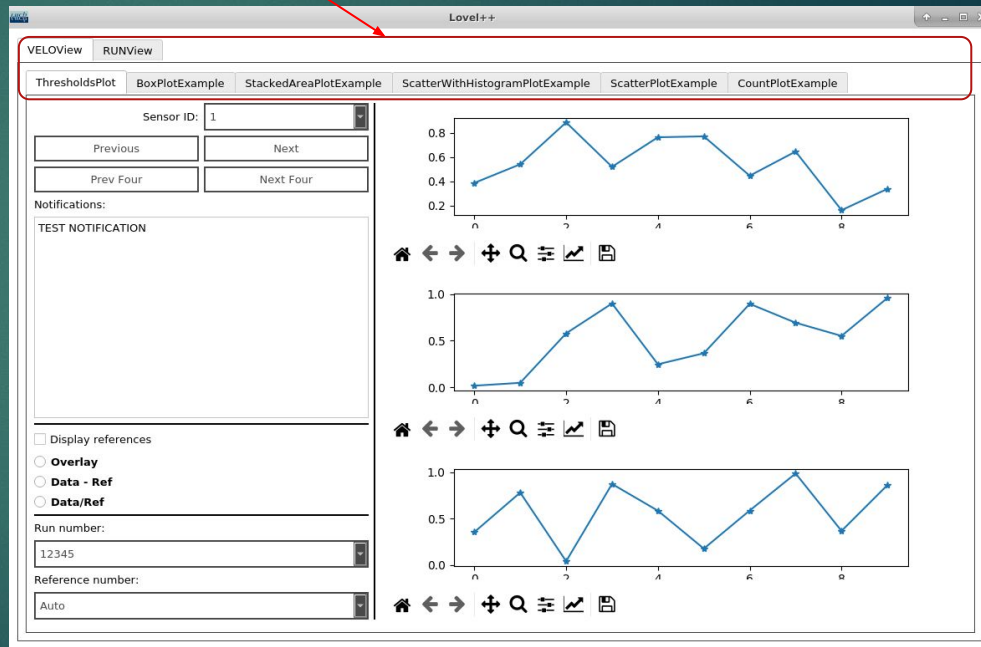


Demo

We can make multiple levels of tabs, as many as you want.

This is to separate things for viewing.

Tabs

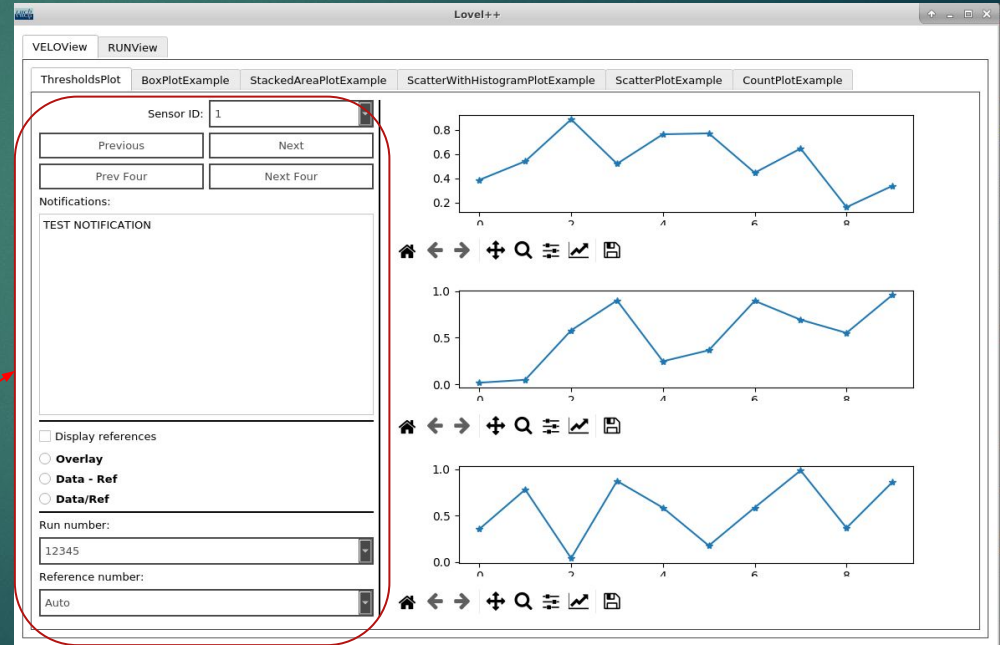


Demo

23

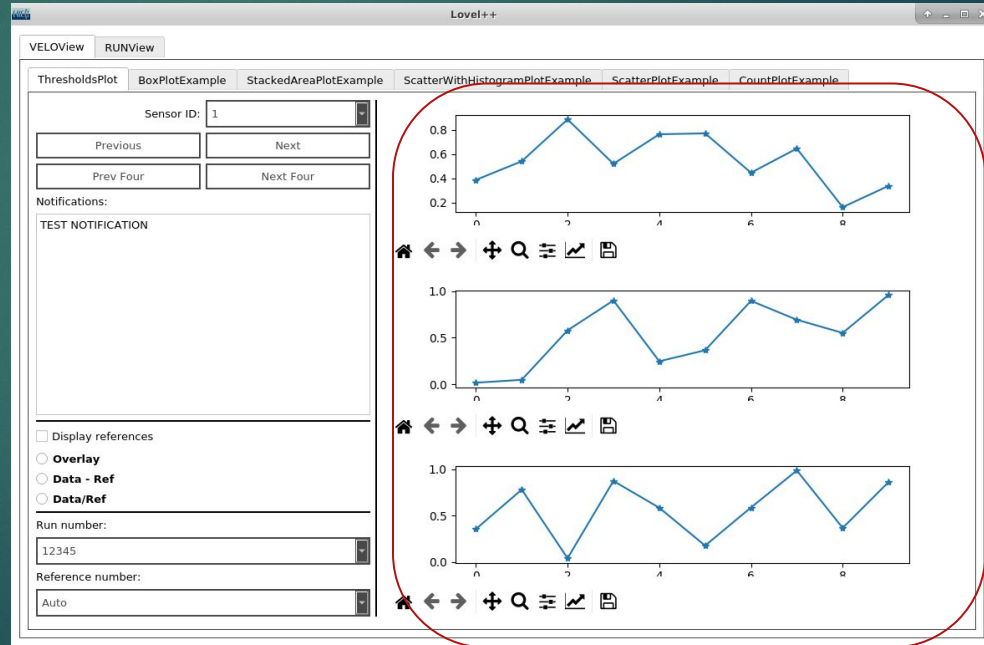
Sidebar - this is actually optional, helps to move through data (like modules, parts, dates etc)

Sidebar



Demo

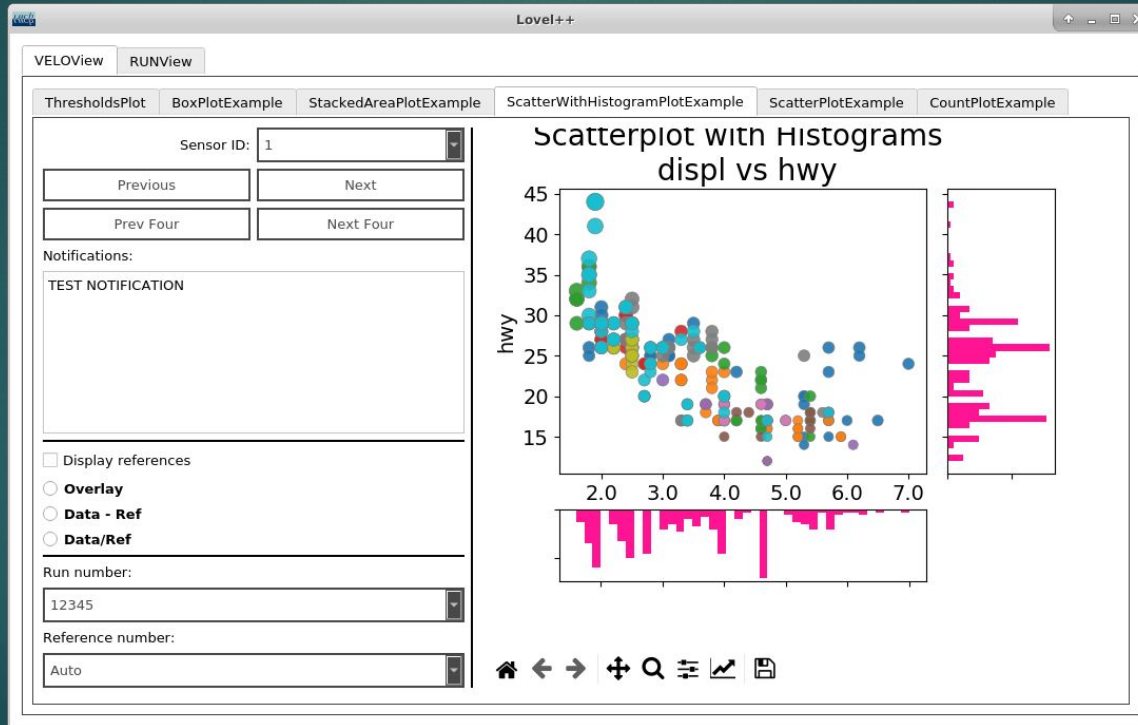
Plots, here you can put anything that you want to view. Actually can be anything that PyQT can take



Plots

Exemplary plot

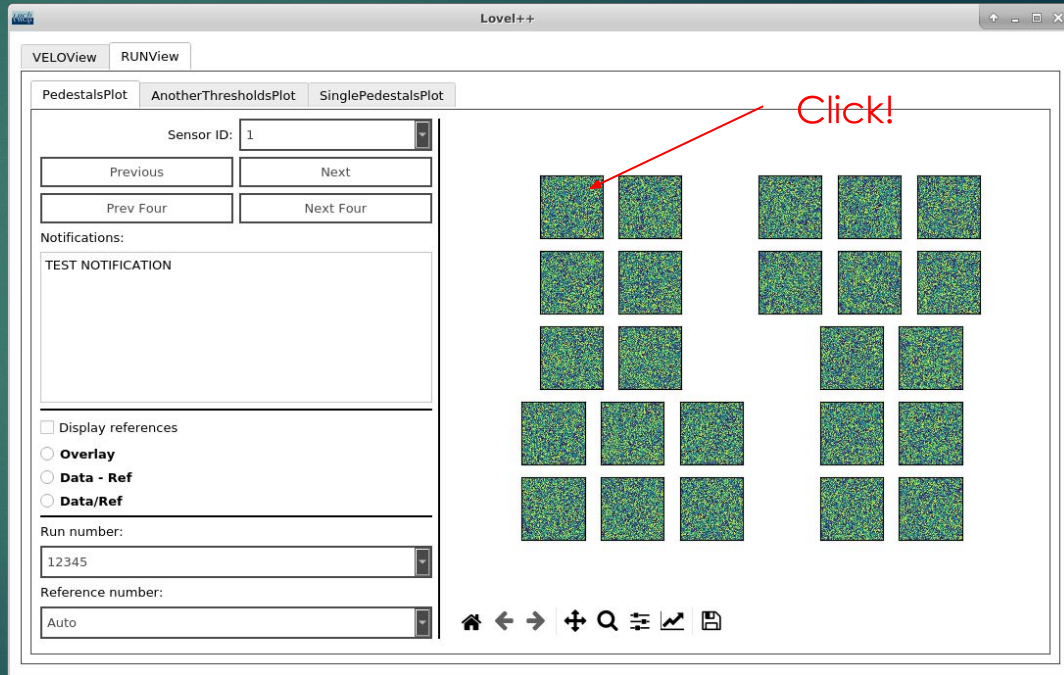
25



Exemplary plot

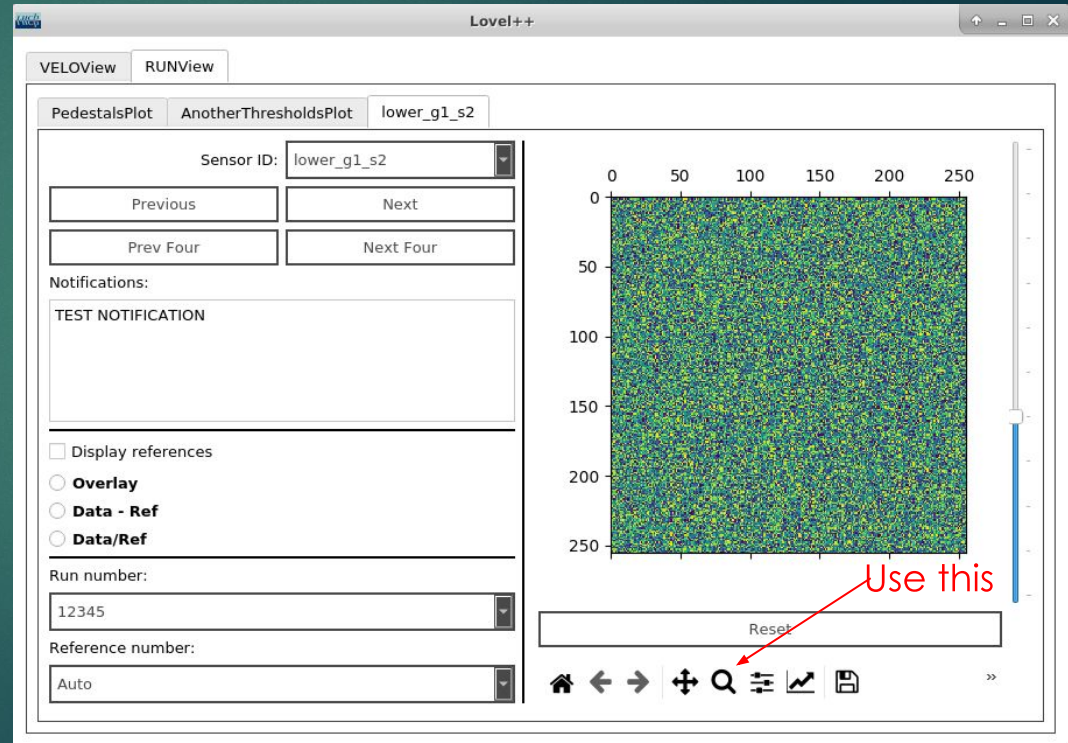
We even tried some interactive options.

Clicking on any of the tiles will get you to another tab



Exemplary plot

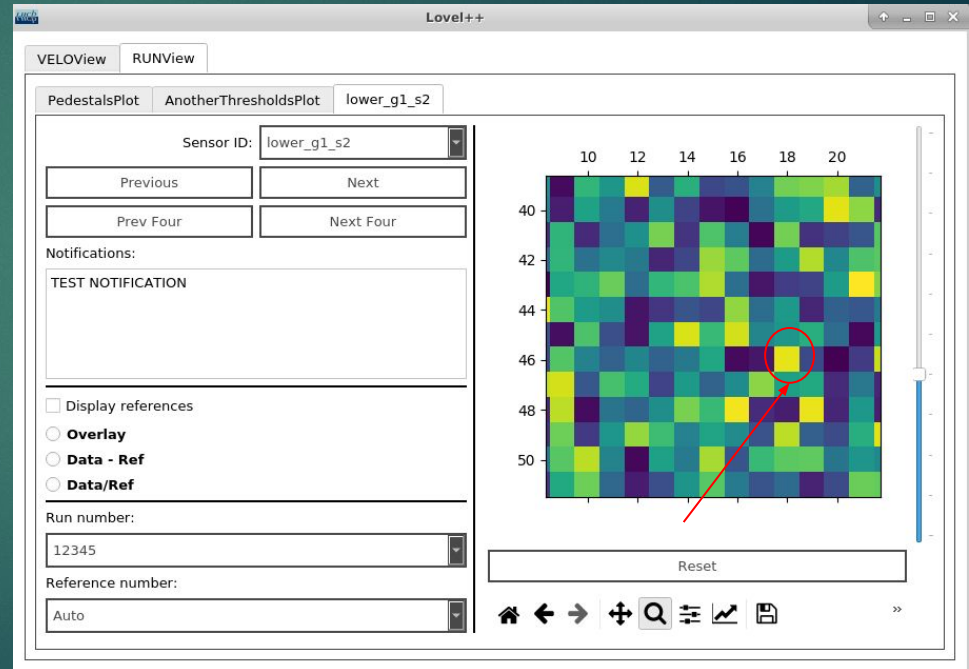
27



Exemplary plot

28

And we can even edit (only in gui, it is not saved anywhere yet)



Exemplary plot

And we can even edit (only in gui, it is not saved anywhere yet)

