Documentation for sTGC plotter application

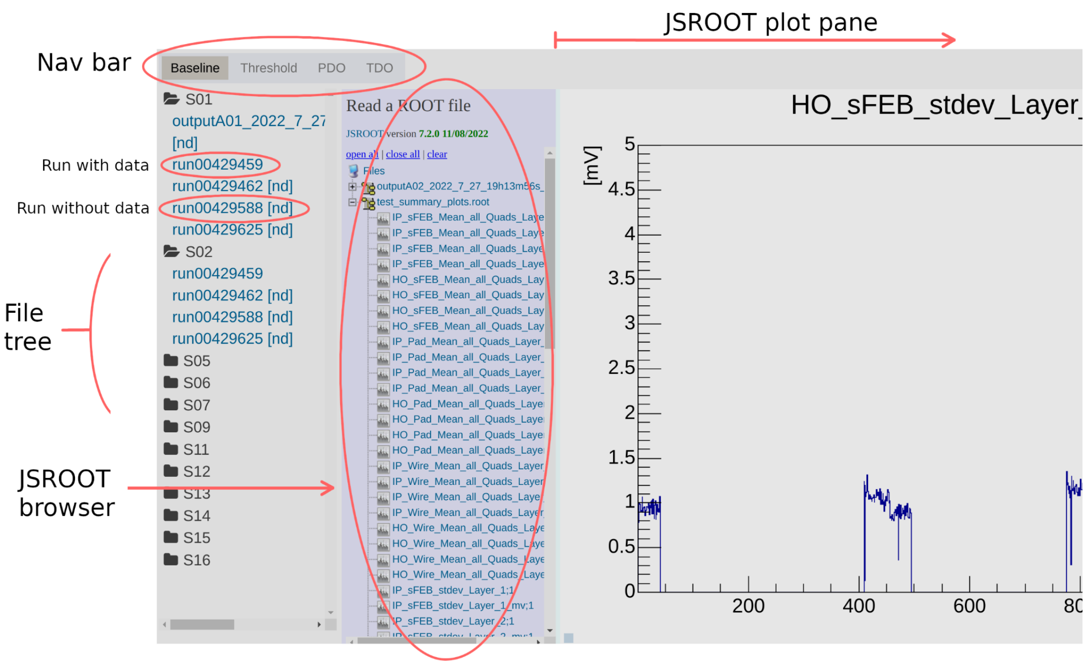
Griffin Kowash
gkowash@gmail.com
October 2022

The plots-stgc application is designed to provide a convenient interface for viewing data generated by the calibration of the New Small Wheel small strip thin gap chambers. This documentation provides instructions for using the application both client-side and server-side, as well as information regarding the project that may be useful for future development. Please feel free to contact me with any questions at the email provided above, and I will do my best to respond promptly.

Client-side use

Navigation homepages

Opening the application presents the user with the "Baseline" homepage. The image below shows the appearance of the page during use with key features labeled.



Nav bar — menu used to navigate between homepages for different plot types (baselines, threshold, PDO, and TDO). Clicking on a tab will redirect the user to another page with an identical layout but different .root files.

File tree — collapsible directory navigator for accessing run data.

Runs — links that have the following properties:

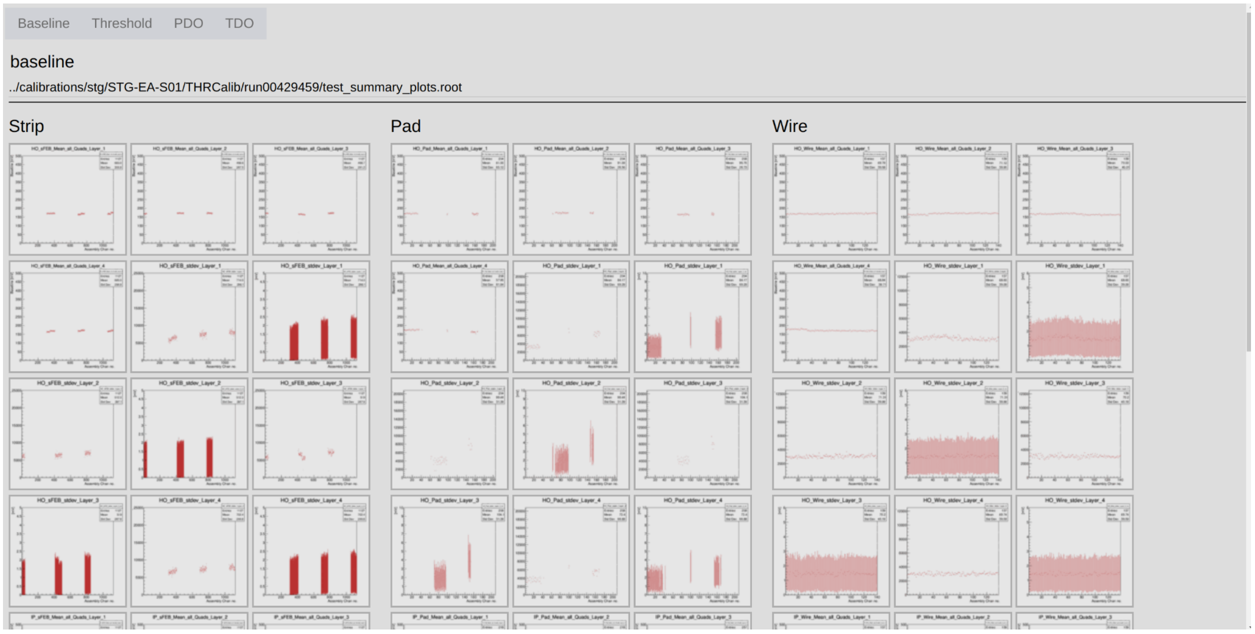
- **Clicking** on a run will redirect the user to the corresponding plot page.
- **Dragging and dropping** a run link onto either of the JSROOT panes will load the corresponding .root file into the JSROOT browser
- **Suffix "[nd]"** ("no data") indicates that there is a folder with that run number present in the directory, but it does not contain a .root file for the plot type currently being viewed. Links with no data will not redirect to a page or load a .root file.

JSROOT interface — allows contents of .root files to be viewed without leaving the navigation homepage. Limited interaction with plots is also supported.

- **JSROOT browser** displays the contents of any .root files that have been loaded via dragging and dropping. Clicking on a link will display the plot in the plot pane.
- **JSROOT plot pane** shows a plot selected from the browser. It supports some real-time interaction:
    - User can zoom in or out like a normal TBrowser and reset the view by double clicking.
    - Plot elements such as titles and legends can be repositioned.
    - The right-click menu offers several plot options, including log axes, style adjustments, and saving the modified plot as a .png or .svg.

Plot pages

Clicking on a run link in the file tree that does not have an "[nd]" suffix will redirect the user to the corresponding plot page. The exact layout may vary depending on the directory structure of the .root file, but will generally appear as shown below:



baseline

../calibrations/stg/STG-EA-S01/THRCalib/run00429459/test_summary_plots.root

Nav bar — same function as described above.

Header — displays plot type and path to the .root file being viewed.

Plot viewer — plots extracted from the .root file.

- Plots are organized into three columns for strip, pad, and wire.
- Clicking on a plot will load a Highslide interface with the following features:
    - Displays an enlarged, higher-resolution version of the plot.
    - Allows neighboring plots from the same column to be viewed using the left and right arrow keys.
    - Can be minimized again by clicking on the plot pane.

When done viewing the plots, the user can either use the nav bar to navigate to a homepage or return to the previous page using their browser.

Server-side use

Updating for new data

Syntax (update_plots.py)

All HTML files and plot images used by the application are generated in advance, so the EOS server files must be updated for any new data to be visible.

The most convenient way to do this is via the Python 3 script *update_plots.py* in the `monitor_server` directory. Executing this script will open one or more .root files, save image files for all plots they contain, create an HTML file to display the plots in the application, and update the navigation file trees to include the new files.

The script can take one of three options to specify the operating mode:

-n — executes only for new .root files without existing plot page files. (Default)

-a — executes for all .root files (new and old), overwriting any previously-generated plot page files.

-f FILEPATH — executes only for a file specified by the user. (This mode currently supports just one file at a time, but could be modified to accept multiple files without too much difficulty.)

The user can also specify a plot type or have the script execute for all four:

-t TYPE — executes only for the provided plot type (baseline, threshold, pdo, tdo), or for all four if using the plot type "all".

Finally, if the Highslide directory is moved from its default location in the monitor_server folder, the user can specify an alternate path:

-j PATH — specifies highslide path for root2html_stgc.py (include up to /highslide-5.0.0/highslide)

An option ***-p PATTERN*** is also present in the utility script *root2html_stgc.py* to allow the use of regex patterns to filter TCanvases within each .root file, but it is currently not accessible via *update_plots.py*. If desired, this feature can be added without much difficulty by referring to the in-code documentation for *root2html_stgc.py*.

Examples

Only new files, all plot types (defaults):

```
python3 update_plots.py
```

New and old files, all plot types:

```
python3 update_plots.py -a
```

Only new files of type "pdo":

```
python3 update_plots.py -t pdo
```

Only the file located at [dirpath]/test_summary_plots.root:

```
python3 update_plots.py -f
      ../calibrations/stg/STG-EA-S01/THRCalib/run00429459/test_su
      mmary_plots.root
```

Only new files, all plot types, Highslide located in directory "monitor_server/util":

```
python3 update_plots.py -j util/highslide-5.0.0/highslide
```

New and old files of type "threshold", Highslide located on OpenShift server:

```
python3 update_plots.py -a -t threshold -j /
```

File descriptions

This section summarizes the function of *update_plots.py* and two other scripts it utilizes, *root2html_stgc.py* and *make_filetree.py*. As of this writing, all three are located in the `monitor_server` directory.

update_plots.py

As described above, this script is the main interface used for updating the application files when new data becomes available. The code searches the `calibrations/stg` directory for .root files matching the provided criteria and executes the main file-generating script *root2html_stgc.py* for each. The script *make_filetree.py* is then executed to include links for the new files in the navigation pages of the application. As *update_plots.py* only serves to make updating more convenient, the user can bypass it and execute the other two scripts manually if desired without any loss of functionality.

root2html_stgc.py

A modified version of *root2html.py*, which was originally written by Ryan Reece, Tae Min Hong, and Alex Tuna. The script accepts a .root file as an argument and uses PyROOT to save each plot as two image files, one for high-resolution viewing and the other for a thumbnail. It then generates an HTML file which uses the Highslide library to present the plots on a webpage.

By default, the directory is generated in the same location and with the same name as the source root file. For example, if the root file is located at

```
stg/STG-EA-S03/THRCalib/run00429459/example_file.root
```

the output directory will be at

```
stg/STG-EA-S03/THRCalib/run00429459/example_file
```

If the user wishes the plot page files to be output to a different directory than that of the .root source file, the `dirname` attribute of class `HighSlideRootFileIndex` can be modified, although further modifications to the script may be required as well. In addition, the script *make_filetree.py* (described below) will need to be updated to reflect the new directory location.

The principal changes made to the original file *root2html.py* are the following:

- Updated to Python 3 (original written in Python 2)
- HTML output page organizes plots by pad, strip, and wire and includes a nav bar so the navigation homepages can be accessed.

Some other minor functional and cosmetic changes are documented in the code as well.

The modified code is designed to make recovering the original functionality fairly straightforward if desired, as major changes can be toggled on or off via an in-code variable "mod". However, some modification would be needed to make this option accessible, and the original functionality has not been recently verified.

make_filetree.py

This script generates the file trees used in the navigation homepages to access run data. Each plot type (baseline, threshold, pdo, tdo) has its own file tree in the `monitor_server/html` folder with the file name *filetree_[type].html*.

Application setup

The following procedure can be used in case the application needs to be recreated from scratch.

Step 1: New Project

Create a new project by following Step 1 ("Getting Started") in the CERN PaaS documentation.

Step 2: Deploy S2I App

Follow instructions in Step 2 ("Deploy Application") → "Deploy From Git Repository" → "Deploy S2I App" in the CERN PaaS documentation.

- Builder image: NGINX
- Source: [repository] (current as of October 2022)
    - Deploy token should not be required if using the above repository.

Step 3: EOS Filesystem

Follow instructions in Step 3 ("Storage") → "EOS Filesystem"

- Credentials for the service account "stgcplot" are required.
- Note that an NGINX configuration file must be included in the repository in order to access EOS files using HTML. (No action needed if already present; refer to the OpenShift server section if missing.)

Other notes

Information is accurate as of October 2022.

Assumed file structure

As currently written, the Python scripts only function with a specific directory hierarchy for the plot data. If this structure changes, or the assumptions made in their writing are incorrect, *update_plots.py* and possibly *make_filetree.py* will need to be modified.

It is currently assumed that .root files for all plot types will be located together in directories of the following form, relative to the `monitor_server` folder:

```
../calibrations/stg/STG-EA-S[##]/THRCalib/run[########]
```

The script *make_filetree.py* is particularly sensitive to the directory structure, as it identifies run folders based on the number of directories in the file path while searching through the `stg` directories. If the folder hierarchy is changed, this algorithm will need to be modified, and ideally made more flexible.

Identifier strings

Files corresponding to each plot type are identified by a unique **identifier string** used in their file names. For example, the file

```
outputA02_2022_7_27_19h13m56s_pdo_plots.root
```

is recognized as a PDO file by the presence of the substring "pdo_plots.root" in the name.

As of this writing, the appropriate identifier strings are not known, and temporary values have been assigned for the purpose of testing based on the .root files available:

Baseline — *summary_plots.root*
Threshold — *outputcanvas.root*
PDO — *pdo_plots.root*
TDO — *summary_plots.root* (repeated; only three files available as of this writing)

Identifier strings are specified in the dictionary `file_key` in both *update_plots.py* and *make_filetree.py*. To change these values, **both** files will need to be updated in order to function properly. Alternatively, the dictionary may be moved to a separate file and imported by each script so only a single file needs to be modified.

EOS server

As of October 2022, the majority of files used by the application are housed in the following EOS directory:

```
/eos/atlas/atlascerngroupdisk/det-nsw-stgc/P1/monitor_server
```

The directory currently has the following structure:

```
monitor_server
    update_plots.py
    root2html_stgc.py
    make_filetree.py
    html
        nav.html
        baseline.html
        threshold.html
        pdo.html
        tdo.html
        filetree_[baseline/threshold/pdo/tdo].html (four files)
    css
        style.css
    JavaScript
        browser_loader.js
        jquery-3.6.0.js
        menu.js
    highslide-5.0.0
        [Highslide files]
    data
        [currently unused]
```

As noted above, the `data` directory is currently not in use. Instead, .root files and any associated image or HTML files generated by *root2html_stgc.py* are located in directories of the form

```
/eos/atlas/atlascerngroupdisk/det-nsw-stgc/P1/calibrations/stg/STG-EA-S
    [##]/THRCalib/run[########]
```

where `[##]` is a number in the range 01–16 and `[########]` is an 8-digit identifier for the run.

In general, file paths in the HTML are dependent on this hierarchy; if it is changed, the HTML navigation pages, HTML strings in *root2html_stgc.py*, and other files may need to be modified to reflect the new structure.

OpenShift server

OpenShift requires two files in the Github source repository in order for the application to function properly.

The first is *index.html*, which is the default page opened when the user accesses the application. The only content is a simple redirect to `monitor_server/html/baseline.html` in the head tag:

```
<meta http-equiv="refresh" content="0; URL=/html/baseline.html"/>
```

The other file is *eos.conf*, which **must** be located in the directory `nginx-default-cfg` within the repository. This file is used to configure NGINX during the application build to allow access to files on EOS, given that the proper credentials are in place (Step 3 of "Application setup").

The contents of *eos.conf* provide aliases for various EOS directories that can be used in all HTML files. The following shows the general formula used to define an alias and the aliases currently in place as of October 2022:

```
location /[name]/ {
      alias /eos/[rest of path]/;
}
```

/home	/eos/ … /monitor_server
/stg	/eos/ … /calibrations/stg
/html	/eos/ … /monitor_server/html
/css	/eos/ … /monitor_server/css
/JavaScript	/eos/ … /monitor_server/JavaScript
/highslide	/eos/ … /monitor_server/highslide-5.0.0/highslide

with … substituted for the path `atlas/atlascerngroupdisk/det-nsw-stgc/P1` for brevity.

If *eos.conf* is modified, the application must be rebuilt via the OKD developer interface in order for the changes to take effect.

Security

For the security of the EOS filesystem, aliases in *eos.conf* should not be linked to directories that contain sensitive files in any of their subpaths, as they may be readable by client-side users. In particular, an alias such as

```
location /atlas/ {
      alias /eos/atlas/;
}
```

would expose the entirety of the ATLAS EOS filesystem, which would be a violation of the CERN Computing Rules.

Contact for questions

If any further clarification on the topics discussed here is required, or if an important point has been omitted from the documentation, please feel free to contact me (Griffin Kowash) at gkowash@gmail.com, and I will do my best to respond promptly.