Thorough-ish summary of what I have worked on

Leila Freitag

Thanks to Andreas, Moritz, Jonas and Beatriz ☺

DESY Summer Student Program 2022

https://github.com/leilafreitag/leila-DESY-scripts

Emittance scan analysis plots for early Run-3

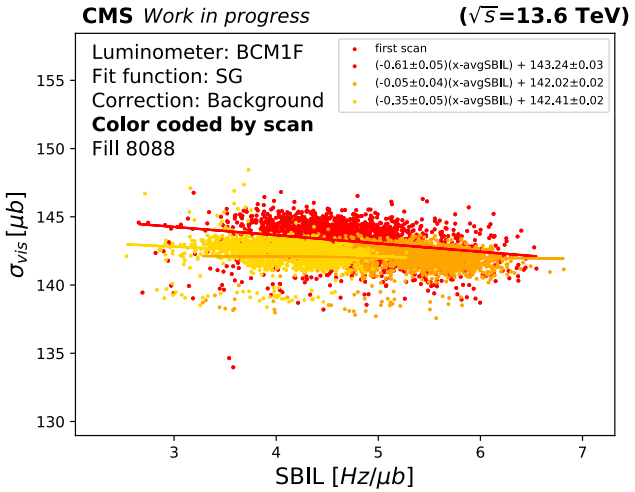
- Learned how to run the VdM Framework to generate up-to-date fit_and_lumidata.csv and fit_and_lumidata_averageBCIDs.csv files
- Made a copy of the framework for myself in /localdata/lfreitag/VdMFramework on brildev1
- Was given a makeplots.py script from Alessia to make some basic plots from the csv
- Used this as a basis to make different plots from the csv file
  - Sigvis vs scan (stability plot)
  - Sigvis vs SBIL (linearity)
  - Plots looking at train structure (sigvis vs SBIL with color coded leading bunches, 2^{nd} 3^{rd} in train, sigvis vs BCID)
  - …
- Santeri set up a folder at https://vdmoutput.web.cern.ch/leilas_plots/ where I gathered the plots I produced

Emittance scan analysis plotting scripts

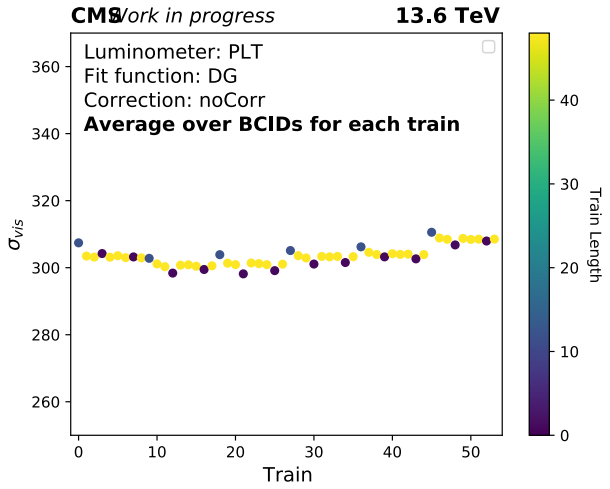
Plotting data from either the fit_and_lumidata.csv or fit_and_lumidata_averageBCIDs.csv file

Script	Plot
makeplots_current_version.py	*sigmavis_vs_SBIL_color_coded_scans* *sigmavis_vs_train* *sigmavis_vs_trainlength*
makeplots_trains.py	*sigmavis_vs_SBIL_color_coded_trains* *sigmavis_vs_BCID_color_coded_trains*
makeplots_leading_bunches.py	*sigmavis_vs_scan_leading_bunches* *sigmavis_vs_train_leading_bunches*
makeplots_from_averageBCIDs.py	*sigmavis_vs_scan* *sigmavis_vs_scan_scaled*
makeplots_comparison.py	*sigmavis_vs_scan_scaled_comparison_detectors* *sigmavis_vs_scan_scaled_comparison_fits*
makeplots_bcids.py	*sigmavis_vs_SBIL_color_coded_bcid*

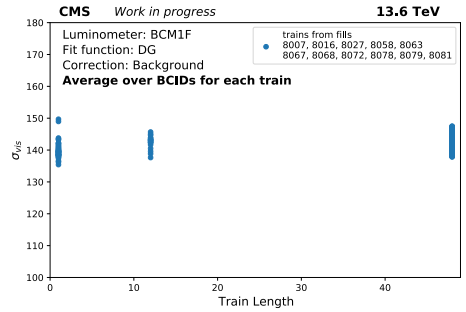
sigmavis_vs_SBIL_color_coded_scans



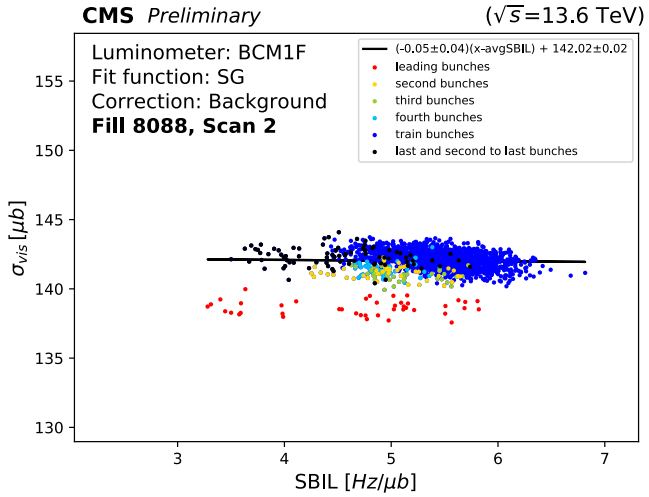
sigmavis_vs_train



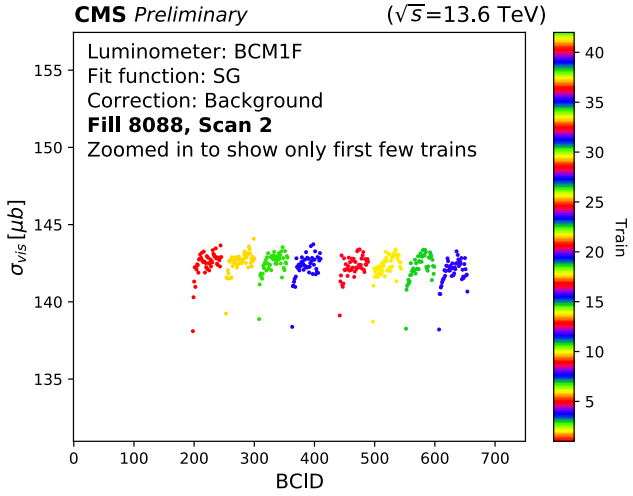
sigmavis_vs_trainlength



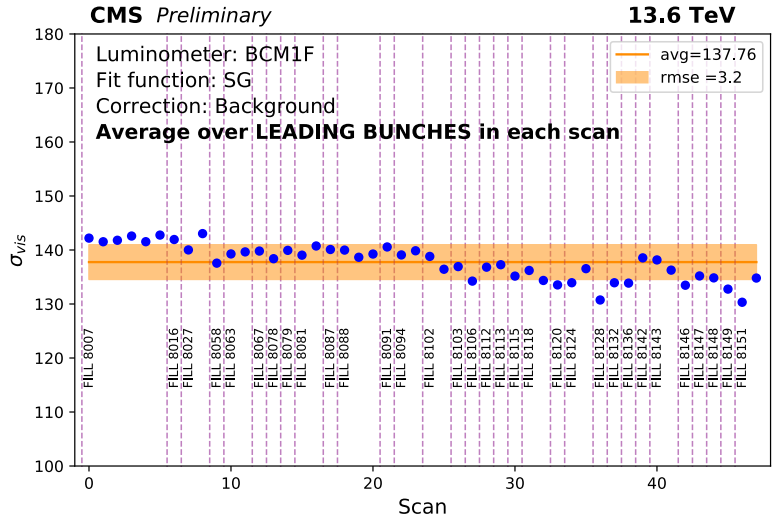
sigmavis_vs_SBIL_color_coded_trains



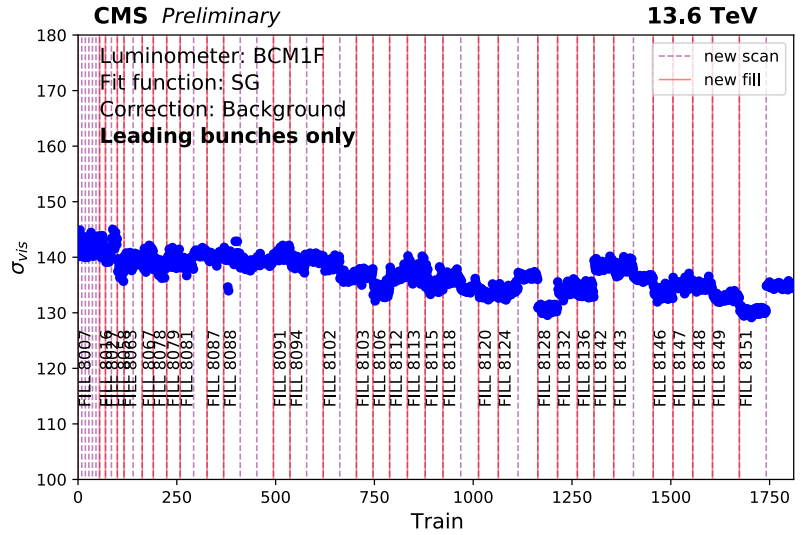
sigmavis_vs_BCID_color_coded_trains



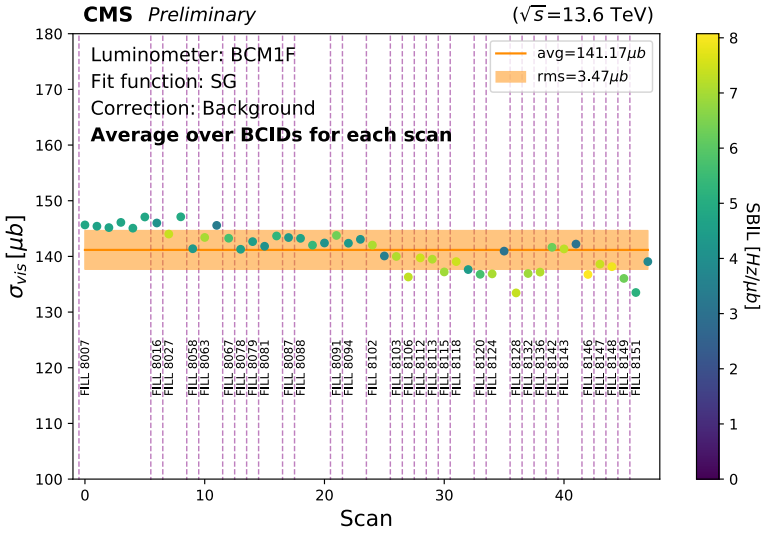
sigmavis_vs_scan_leading_bunches



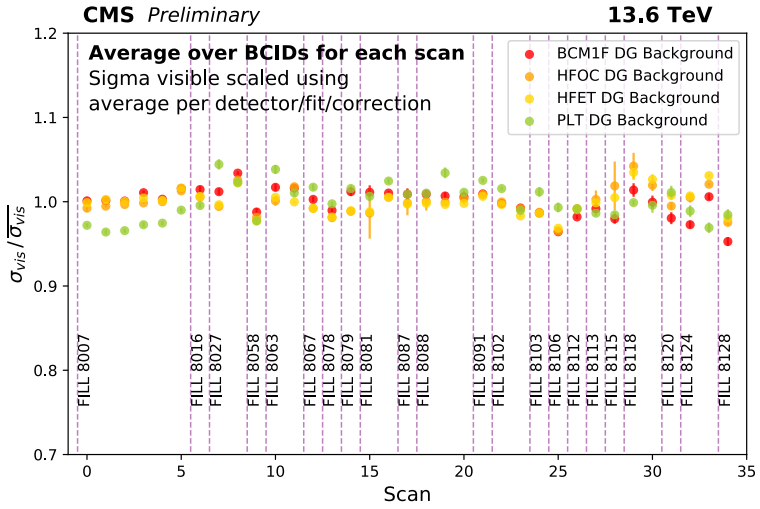
sigmavis_vs_train_leading_bunches



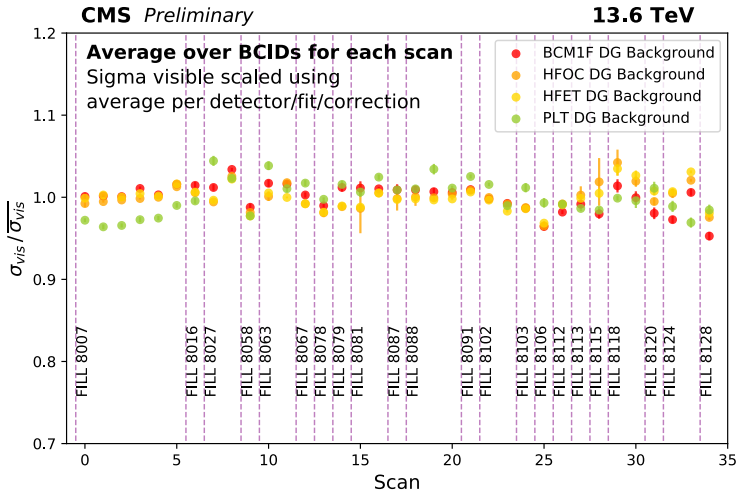
sigmavis_vs_scan



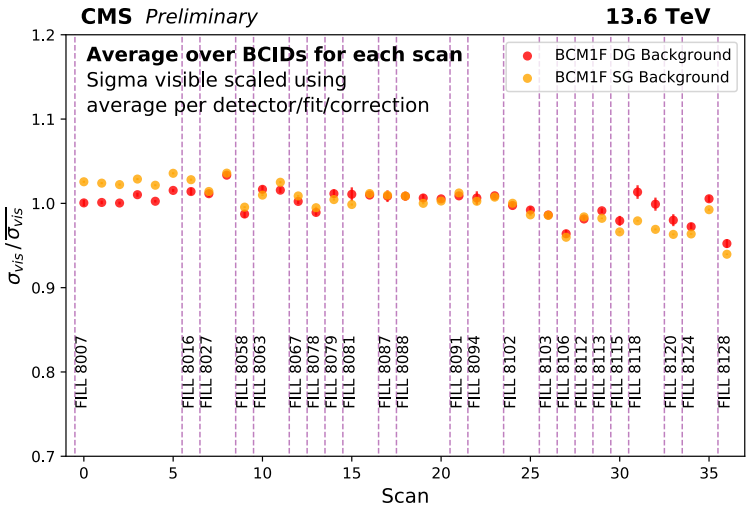
sigmavis_vs_scan_scaled



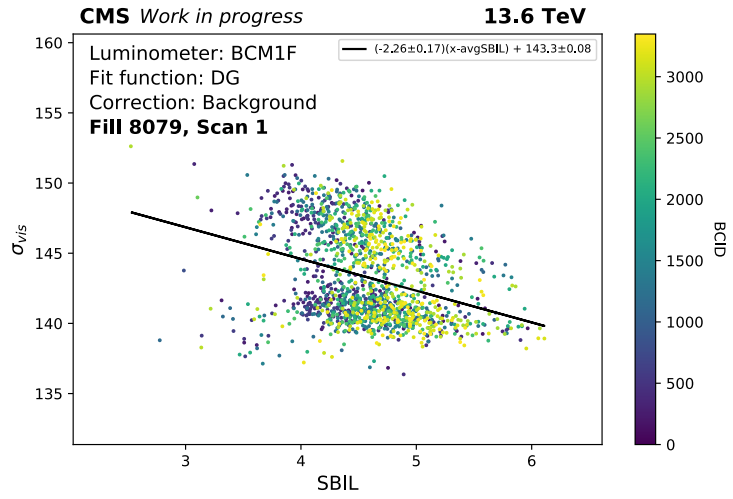
sigmavis_vs_scan_scaled_comparison_detectors



sigmavis_vs_scan_scaled_comparison_fits



sigmavis_vs_SBIL_color_coded_bcid



Investigating the albedo correction for BCM1F

- Learned from Moritz and Jonas how to reprocess fills with for example different albedo correction, channel mask…

Q1: Does the fact that we measure lower sigma visible for leading bunches than train bunches have anything to do with the albedo correction?

Q2: Understand how data-driven albedo model is produced, apply a new channel mask, and reprocess. Does improved channel mask reduce the overcorrection of the albedo model in the first empty bunches after bunch trains?

(Q1) Albedo correction's effect on leading bunches

- Wanted to see if changing the albedo correction could improve the discrepancy seen between the measured sigma visible for leading bunches vs train bunches (leading bunches have lower sigma visible)
- To reprocess and then run the VdM Framework on a fill, used Moritz's processor.py and repack_for_VdM.py scripts
- Tried the following, and saw no significant change:
  - Removing the correction entirely (do_albedo_correction = false)
  - Changing how often the correction is recalculated (calculate_albedo_every = 7, 1,4,70 [xnb4])
  - Changing the queue length, or how many histograms/timesteps are used to calc the correction (albedo_queue_length = 150, 600)

→The albedo correction makes no significant impact on the lower sigma visible of leading bunches

(Q2) Reprocessing with new channel mask

- Wanted to understand how data-driven albedo model is produced, apply a new channel mask, and reprocess to see if this changes the overcorrection?
- Reproduced Moritz' current albedo model using fill 7921 (and 7920) (consists of 2 single bunches)
  - Take list of rate of all empty bunches after the single bunch, and normalize each by the rate measured in the single bunch. Then subtract the constant baseline (which is I think noise) from each element. This is the model.
- (unimportant or unsuccessful) Tried to make models using other fills
  - 7923 (fill with 8 single bunches)
  - 8088 (fill with a single bunch among lots of bunch trains)
  - Fitted with exponentials to subtract residual contributions from bunches before the leading bunch to get models
- Updated channel mask
  - Got an updated channel mask from Jonas, reprocessed fill 7921 with this new channel mask and no albedo correction, determined the new albedo correction. Then reprocessed fill 8088 with the new channel mask and the new albedo correction and compared it to the original mask with original correction.
  - Result of comparison: first empty bunch after train is only slightly less overcorrected

Reprocessing and albedo scripts

Script	Description
processor.py	Script to reprocess fills from Moritz. Fixed tiny bug (the last row in the original hd5 file was previously missing in the reprocessed file)
make_albedo_correction.py	making albedo correction from the fill 7920 or 7921, that had 2 single bunches. Currently makes correction for both old and new channel mask.
make_albedo_analysis.py	Script to look at the behavior of the empty bunches immediately after colliding bunches, to see how the albedo correction is doing.
lookatAlbedo.py	Based on script from moritz, used to look at hd5 files and compare reprocessed data to original processed data (from brildata)