

Monitor of the trigger efficiency

Mirei AOYAMA

Nara Woman's university

2019-08-28

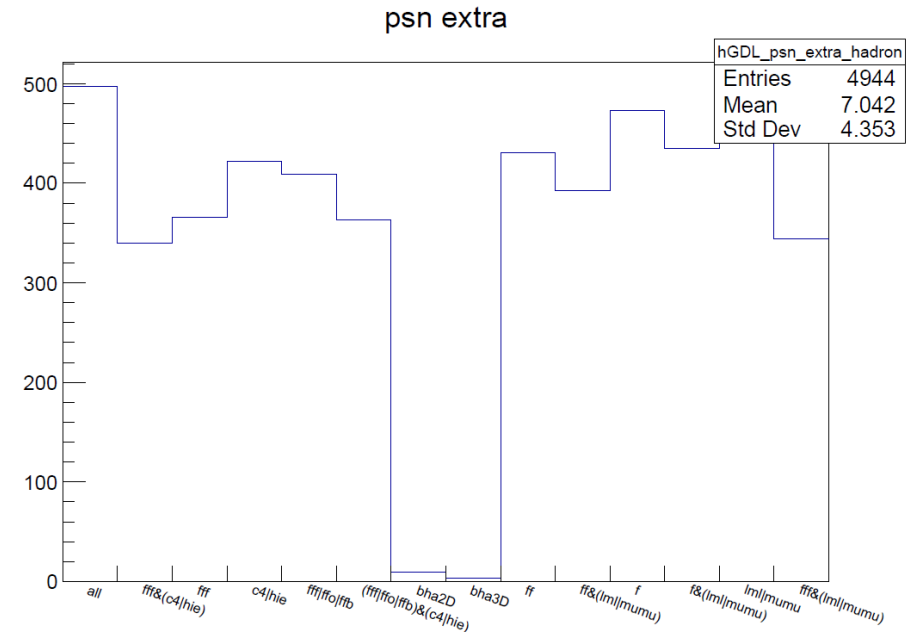
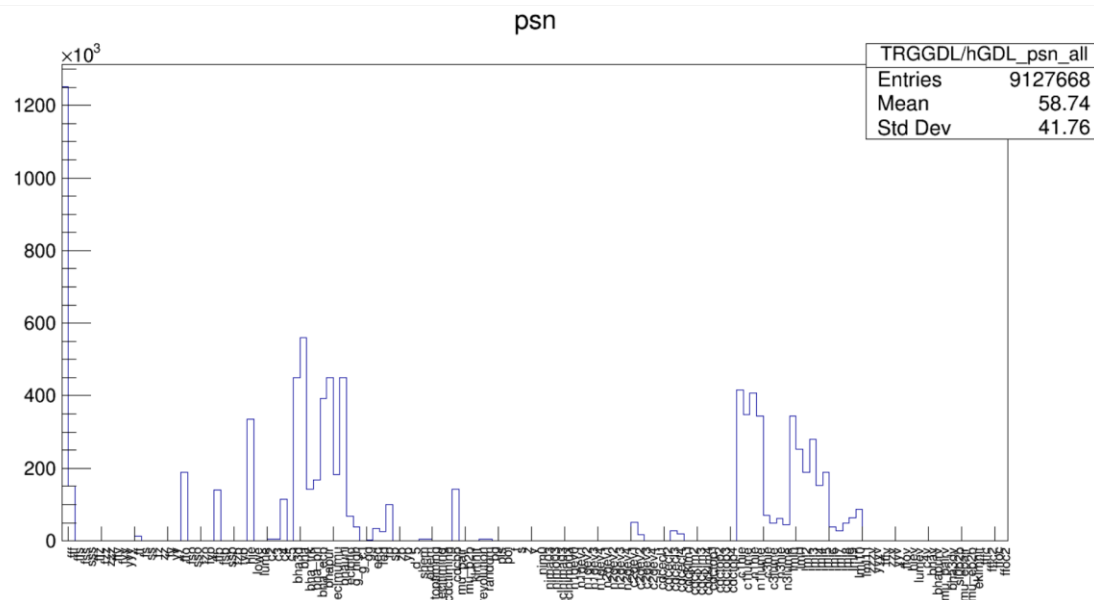
Purpose

Prepare online monitor of various efficiencies as function of the run number.

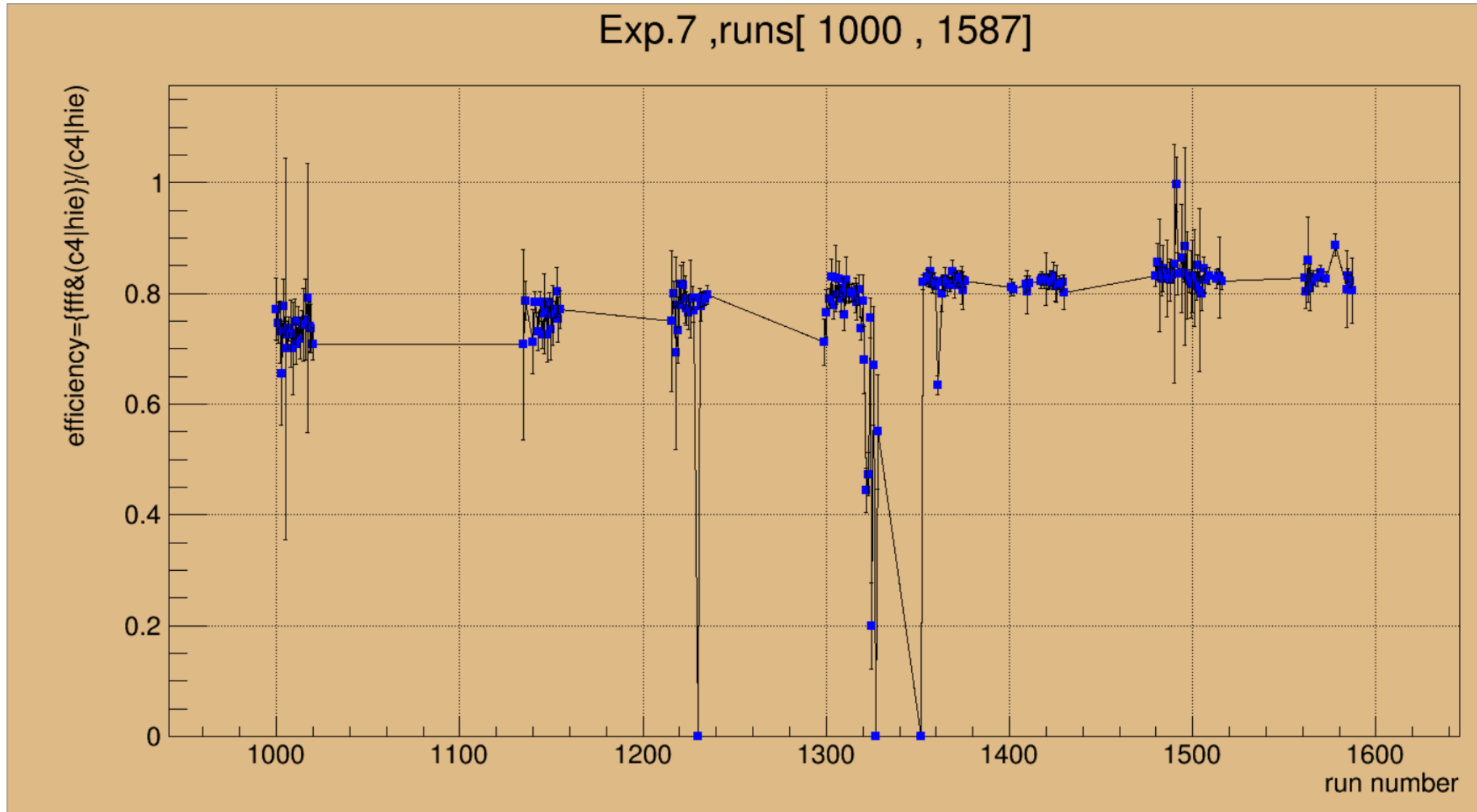
So far, I did following things.

1. Confirm that I can access the root file of the online DQM.
2. Get the value (fff,ffo,hie,c4,bhabha etc) in the “psn” histogram by Pyroot.
3. Prepare a Python script to scan DQM root files for given experiment and run-ranges.
4. Make an efficiency plot as a function of run by using TGraphErrors.

Definition of fff efficiency for hadronic event = $\frac{\text{fff} \&(\text{c4}|\text{hie})}{\text{c4}|\text{hie}}$.



fff efficiency for hadronic event = $\frac{\text{fff} \&(\text{c4|hie})}{\text{c4|hie}}$ vs run number



What to do next

1. Put it on the Web.
2. Prepare various kind of monitor other than fff efficiency for hadronic event $= \frac{\text{fff} \&(\text{c4}|\text{hie})}{\text{c4}|\text{hie}}$.
3. Update plot automatically.

Plans

1. A plot will be prepared in September.
2. It will be updated using the ARICH framework of the year.

Back up

Python file for preparing an efficiency prot.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

#from ROOT import * <--
from ROOT import TFile,
TKey,TH1D,TCanvas,TGraphErrors
#
import os,re,sys
import glob,math
from array import array

#print initial parameters
arg = sys.argv
if len(arg) != 4:
    print("No of argument should be 3")
    print("example ./file_saneff.py 7 1000
1587")
    sys.exit(-1)
exp_num = int(sys.argv[1])
first_run= int(sys.argv[2])
last_run= int(sys.argv[3])

os.chdir(f"/gpfs/fs02/belle2/users/taichi
ro/dqm/e000{exp_num}")

file_list =sorted(glob.glob('*run_all'))
efflist=[]
b11=[]
b33=[]
nfile=0
run=[]
b1es=[]
b3es=[]
effelist=[]
for data in file_list:
    r=re.findall(r'¥d{5}',data)
    u=int(r[0])
    #print(ru)
    if u >= first_run and u <= last_run:
        run.append(u)
        os.chdir("%s"%data)
        f = TFile("merge.root")
        t=f.Get("TRGGDL")
        h=t.Get("hGDL_psn_extra_hadron")
        #b0=all,b1=fff&(c4|hie),b2=fff,b3=c4|hie
        b1=h.GetBinContent(2)
        b2=h.GetBinContent(3)
        b3=h.GetBinContent(4)
        b11.append(b1)
        b33.append(b3)
        b1e=h.GetBinError(2)
        b3e=h.GetBinError(4)
        #b1es.append(b1e)
        #b3es.append(b3e)
        if b1==0 or b3==0:
            efflist.append(0)
            effelist.append(0)
        else:
            efficiency=b1/b3
            efflist.append(efficiency)
            effe=efficiency*(math.sqrt((b1e/b1)**2
+ (b3e/b3)**2))
            effelist.append(effe)
            os.chdir("../")
            #print(run)
            #print("end of run loop. number of
run="+str(len(run)))
            #Draw a figure
            c1 = TCanvas("c1","efficiency vs
runnumber",200,10,700,500)
            c1.SetFillColor(42)
            c1.SetGrid()
            c1.GetFrame().SetFillColor(21)
            c1.GetFrame().SetBorderSize(12)

            runs=array('f',run)
            eff=array('f',efflist)
            n=len(run)
            rune=array('f',[0]*n)
            ye=array('f',effelist)

            gr=TGraphErrors(n,runs,eff,rune,ye)
            gr.SetTitle(f"Exp.{exp_num},runs[ {first_r
un} , {last_run}]")
            gr.SetMarkerColor(4)
            gr.SetMarkerStyle(21)
            gr.GetXaxis().SetTitle("run number")
            gr.GetYaxis().SetTitle("efficiency={fff&(c
4|hie)}/{(c4|hie)}")
            gr.Draw("AIP")
            c1.Update()
            input('push Enter')
```

Where the data is

/gpfs/fs02/belle2/users/taichiro/dqm/e0007

This time I used the data of merge.root in the directory with _allrun_all.

```
sshcc1.kek.jp - aoyamami@aau01:~ VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
r01000_all/
r01000_allrun_all/
r01000_allrun_skim_hadron/
r01000_allrun_skim_mumu/
r01000_sumeff_mumu2/
r01001_allrun_all/
r01001_allrun_skim_hadron/
r01001_allrun_skim_mumu/
r01001_sumeff_mumu2/
r01002_allrun_all/
r01002_allrun_skim_hadron/
r01002_allrun_skim_mumu/
r01002_sumeff_mumu2/
r01003_allrun_all/
r01003_allrun_skim_hadron/
r01003_allrun_skim_mumu/
r01003_sumeff_mumu2/
r01004_allrun_all/
r01004_allrun_skim_hadron/
r01004_allrun_skim_mumu/
r01004_sumeff_mumu2/
r01005_allrun_all/
r01005_allrun_skim_hadron/
r01005_allrun_skim_mumu/
r03230_allrun_skim_mumu/
r03230_sraw/
r03230_sumeff_mumu2/
r03231_allrun_skim_hadron/
r03231_allrun_skim_mumu/
r03231_sraw/
r03231_sumeff_mumu2/
r03232_allrun_skim_hadron/
r03232_allrun_skim_mumu/
r03232_sraw/
r03232_sumeff_mumu2/
r03233_allrun_skim_hadron/
r03233_allrun_skim_mumu/
r03233_sraw/
r03233_sumeff_mumu2/
r03234_allrun_skim_hadron/
r03234_allrun_skim_mumu/
r03234_sraw/
r03234_sumeff_mumu2/
r03236_allrun_skim_hadron/
r03236_allrun_skim_mumu/
r03236_sraw/
r03236_sumeff_mumu2/
r03237_allrun_skim_hadron/
```

