Calculate POT

Marina Reggiani-Guzzo (The University of Manchester / The University of Edinburgh)

This code teaches you how to calculate the POT for a given sample. There is an option of applying cuts on the sample before calculating the POT, which is important for beam-on, for example, as we are interested in the POT after applying the good-quality check and in a specific neutrino mode (for run 1 it means to select the runs < 6748). So make sure you are applying all the necessary cuts before calculating the POT, as the POT is used as an input for the analysis.

In [1]:

```
from IPython.core.display import display, HTML # fix window size
display(HTML('<style>.container {width:100% !important;}</style>'))
import pandas as pd
import uproot3 as uproot
import numpy as np

# ==================== #
#   IMPORT FUNCTIONS   #
# ==================== #

import os
import sys
module_path = os.path.abspath(os.path.join('/uboone/app/users/mguzzo/phd_anti
module_path = os.path.abspath(os.path.join('/home/marinaguzzo/Desktop/antinue
if module_path not in sys.path:
    sys.path.append(module_path)

from ipynb.fs.full.antinue_myfunctions import * # declare all functions here,
```

In [2]:

```
def gen_run_subrun_list(input_file, name_list):
    # Get useful variables
    T_eval = uproot.open(input_file)['wcpselection/T_eval']
    df_eval = T_eval.pandas.df(['run', 'subrun'], flatten=False)
    df_eval.drop_duplicates(inplace=True)
    np.savetxt(name_list, df_eval.values, fmt='%d')

# --- keep only good-quality events
def apply_good_quality_check(df_in, file):
    # Create dataframe with the good-quality runs you printed above
    df_good_runs = pd.read_csv(file, sep=", ", header=None, engine='python')
    df_good_runs = df_good_runs.T # transpose so it's a column dataframe
    df_good_runs.rename(columns={0:'run'}, inplace=True) # rename column to m

    # apply the cut
    df = pd.merge(df_in, df_good_runs, how='left', indicator=True)
    df_good = df[df._merge=='both'] # good-quality runs in input sample
    df_bad = df[df._merge!='both'] # bad-quality runs in input sample

    # uncomment if you want to select GOOD-QUALITY events
    df_final = df_good.drop(columns="_merge") # remove column "merge" only us
    df_final.drop_duplicates(inplace=True) # remove duplicates

    bad_quality = False # false=keep good-quality, true=keep bad-quality
    if(bad_quality):
        df_final = df_bad.drop(columns="_merge")
        df_final.drop_duplicates(inplace=True)
    return df_final
```

Run 1 Beam-On

This part is necessary for beam-on only. As we want to apply the good-quality check, we need the list of good-quality runs that will be used as a filter to compare to the runs present in the sample and keep only the good-quality ones. This part of the code you have to do manually on the terminal. First, create all the necessary tickets to run samweb commands:

> source /cvmfs/uboone.opensciencegrid.org/products/setup_uboone.sh
> setup uboonecode v09_32_00 -q e20:prof
> setup sam_web_client
> voms-proxy-init -noregen -rfc -voms fermilab:/fermilab/uboone/Role=Analysis
> kx509

Which can quickly be ran as:

> cd /uboone/app/users/mguzzo/phd_antinue
> source setup_samweb.sh

Then, run the samweb command that will print the list of run/subrun for the good-quality ones:

> cd /uboone/app/users/mguzzo/phd_antinue
> samweb describe-definition goodruns_mcc9_run1_high_lifetime > run1_beamon_goodquality.list

Unfortunately the list generated above is not in the correct format, so let's work on this so it is in the format that the code below needs it to be. You have to do a few things:

- (1) delete the header of the file, everything that is not run numbers
- (2) manually remove all the new lines and make it into a single line list
- (3) make sure all run numbers have ", " in front of it, including the last one

Once you're done with the steps above, you'll have a run1_beamon_goodquality.list in the correct format to be used by the code below:

In [10]:

```
# ===================== #
#   INPUT INFORMATION   #
# ===================== #

# you need:
# 1) the good-quality run list generated above (samdef: goodruns_mcc9_run1_hi
# 2) the checkout file

inFile = run1_datafile

# ======================== #
#   POT OF ENTIRE SAMPLE   #
# ======================== #

# save the original run/subrun list, without applying any cut
T_eval = uproot.open(inFile)['wcpselection/T_eval']
```

```
df_eval = T_eval.pandas.df(['run', 'subrun'], flatten=False)
df_eval.drop_duplicates(inplace=True)
#np.savetxt("/uboone/app/users/mguzzo/antinue/run_subrun_list_run1_beamon_ori

# ======================== #
#   APPLY CUTS ON SAMPLE   #
# ======================== #

df_filtered = apply_good_quality_check(df_eval, 'run1_beamon_goodquality.list
#np.savetxt("/uboone/app/users/mguzzo/antinue/run_subrun_list_run1_beamon_fil

# --- select only FHC events
df_filtered = df_filtered[df_filtered.run < 6748]
np.savetxt("/uboone/app/users/mguzzo/phd_antinue/run1_beamon_goodquality_filt
```

Open a terminal:

> cd /uboone/app/users/mguzzo/antinue
>
> source setup_samweb.sh
>
> unsetup python
>
> setup python v2_7_3
>
> python2.7 getDataInfo.py -v3 --format-numi --prescale --run-subrun-list
>
> run1_beamon_goodquality_filtered.list

```
Read 70595 lines from run_subrun_list_run1_beamon_filtered2.list
           EXT          Gate1         EA9CNT         tor101
tortgt    EA9CNT_wcut    tor101_wcut    tortgt_wcut
    19374300.0     5434744.0     5448638.0     2.024e+20
2.019e+20     5304302.0     2.018e+20     2.014e+20

    EXT_unbiased_PrescaleAlgo
387608.400000
    NUMI_unbiased_PrescaleAlgo
87212.376000
    EXT_NUMIwin_FEMBeamTriggerAlgo
3390539.900000
    NUMI_FEMBeamTriggerAlgo
5434744.000000
    EXT_BNBwin_FEMBeamTriggerAlgo
19374300.000000
```

In [5]:

```
# Run 1 Beam-On
run1_data_pot = 2.01e+20
run1_EA9CNT_wcut = 5304302.0
print("Run 1 DATA POT = %.2e" % run1_data_pot)
```

```
Run 1 DATA POT = 2.01e+20
```

Run 1 EXT

To calculate the EXT POT, first calculate the beam-on variables above.

In [6]:

```
inFile_ext_run1 = run1_extfile
gen_run_subrun_list(inFile_ext_run1, "run1_beamoff_run_subrun.list")
```

Open a terminal:

> cd /uboone/app/users/mguzzo/antinue
>
> source setup_samweb.sh
>
> unsetup python
>
> setup python v2_7_3
>
> python2.7 getDataInfo.py -v3 --format-numi --prescale --run-subrun-list run1_beamoff_run_subrun.list

---

```
Read 41801 lines from run_subrun_list_run1_ext.list
           EXT          Gate1         EA9CNT         tor101
tortgt    EA9CNT_wcut    tor101_wcut    tortgt_wcut
    19489942.0     2334674.0     2340049.0     8.693e+19
8.675e+19     2229613.0     8.661e+19     8.643e+19

    EXT_unbiased_PrescaleAlgo
260105.096000
    NUMI_unbiased_PrescaleAlgo
37315.760000
    EXT_NUMIwin_FEMBeamTriggerAlgo
2466466.930000
    NUMI_FEMBeamTriggerAlgo
2334498.000000
    EXT_BNBwin_FEMBeamTriggerAlgo
19489942.000000
```

In [7]:

```
run1_EXT_NUMIwin_FEMBeamTriggerAlgo = 2466466.93
run1_ext_pot = run1_data_pot/(run1_EA9CNT_wcut/run1_EXT_NUMIwin_FEMBeamTrigge
print("Run 1 EXT POT = %.2e" % run1_ext_pot)
```

```
Run 1 EXT POT = 9.35e+19
```

Run 3 EXT

Get list of good-quality runs for run3b:

```
uboonegpvm04$ samweb describe-definition
goodruns_mcc9_run3_high_lifetime
Definition Name: goodruns_mcc9_run3_high_lifetime
  Definition Id: 77501387
  Creation Date: 2020-04-24T22:14:51+00:00
       Username: uboonepro
          Group: uboone
     Dimensions: defname: goodruns_mcc9_run3_hardcoded and
run_number < 14643 or run_number > 14736 and run_number < 16698
or run_number > 16788
```

> samweb describe-definition goodruns_mcc9_run3_hardcoded > calculate_pot_run3_goodquality.list

In [ ]:

```
inFile_ext_run3 = '/uboone/data/users/mguzzo/wirecell/run3_ext/wirecell_run3_
```

```
# create dataframe with run/subrun
T_eval_ext_run3 = uproot.open(inFile_ext_run3)['wcpselection/T_eval']
df_eval_ext_run3 = T_eval_ext_run3.pandas.df(['run', 'subrun'], flatten=False
df_eval_ext_run3.drop_duplicates(inplace=True)

# apply good-quality runs cut from list created above
df_ext_run3_filtered = apply_good_quality_check(df_eval_ext_run3, 'calculate_

# also apply the remaning cuts by hand to make sure
# "run_number < 14643 or run_number > 14736 and run_number < 16698 or run_num
df_ext_run3_filtered = df_ext_run3_filtered[(df_ext_run3_filtered.run<14643)
                                            ((df_ext_run3_filtered.run>14736)
                                            (df_ext_run3_filtered.run>16788)]

# save the remaining events into a list that will be used to calculate the P
np.savetxt("/uboone/app/users/mguzzo/antinue/run_subrun_list_run3_ext_filtere
```

Open a terminal:

> cd /uboone/app/users/mguzzo/antinue
>
> source setup_samweb.sh
>
> unsetup python
>
> setup python v2_7_3
>
> python2.7 getDataInfo.py -v3 --format-numi --prescale --run-subrun-list run_subrun_list_run3_ext_filtered.list

---

```
ubooneg pvm04$ python2.7 getDataInfo.py -v3 --format-numi --
prescale --run-subrun-list
run_subrun_list_run3_ext_filtered.list
Read 357671 lines from run_subrun_list_run3_ext_filtered.list
              EXT          Gate1         EA9CNT         tor101
tortgt    EA9CNT_wcut    tor101_wcut    tortgt_wcut
    184405418.0     10513907.0     10512474.0      4.823e+20
4.803e+20      9926043.0      4.821e+20      4.801e+20
Warning!! NuMI data for some of the requsted runs/subruns is not
in the database.
1 runs missing NuMI data (number of subruns missing the data):
16228 (1),

     EXT_unbiased_PrescaleAlgo
3786069.160000
     NUMI_unbiased_PrescaleAlgo
168222.512000
     EXT_HSN_c0_FEMBeamTriggerAlgo
36881083.600000
     EXT_BNBwin_2017Dec_SWTrigger5PE_FEMBeamTriggerAlgo
179555601.000000
     NUMI_2017Dec_SWTrigger8_5PE_FEMBeamTriggerAlgo
2404.000000
     EXT_HSN_c1_FEMBeamTriggerAlgo
9357.000000
     EXT_BNBwin_2017Dec_v2_FEMBeamTriggerAlgo
16702.000000
     EXT_NUMIwin_FEMBeamTriggerAlgo
32270948.150000
     NUMI_2018May_FEMBeamTriggerAlgo
```

```
1949338.000000
    EXT_NUMIwin_2018May_FEMBeamTriggerAlgo
14450970.800000
    NUMI_FEMBeamTriggerAlgo
10513907.000000
    EXT_BNBwin_2017Dec_v1_FEMBeamTriggerAlgo
16611.000000
    EXT_BNBwin_2017Dec_v3_FEMBeamTriggerAlgo
16703.000000
    NUMI_2017Dec_SWTrigger9PE_FEMBeamTriggerAlgo
1294.000000
    EXT_BNBwin_FEMBeamTriggerAlgo
184405418.000000
    EXT_NUMIwin_2017Dec_SWTrigger8_5PE_FEMBeamTriggerAlgo
2637.775000
    EXT_NUMIwin_2017Dec_SWTrigger9PE_FEMBeamTriggerAlgo
2855.650000
    EXT_BNBwin_2017Dec_v4_FEMBeamTriggerAlgo
15807.000000
    NUMI_2017Dec_SWTrigger8PE_FEMBeamTriggerAlgo
2434.000000
    EXT_NUMIwin_2017Dec_SWTrigger8PE_FEMBeamTriggerAlgo
2693.950000
```

Observation: there are more lines for the Run3b output in comparison to Run1 due to the trigger changes that happened during Run3b. So don't worry about that.

Summary

In [8]:

```python
# Run 1 Beam-On
run1_data_pot = 2.01e+20
run1_EA9CNT_wcut = 5304302.0
print("Run 1 DATA POT = %.2e" % run1_data_pot)

# Run 1 EXT
run1_EXT_NUMIwin_FEMBeamTriggerAlgo = 2466466.93
run1_ext_pot = run1_data_pot/(run1_EA9CNT_wcut/run1_EXT_NUMIwin_FEMBeamTrigge
print("Run 1 EXT POT = %.2e" % run1_ext_pot)
```

```
Run 1 DATA POT = 2.01e+20
Run 1 EXT POT = 9.35e+19
```