Analysis Code Planning

Stephen Kay
University of Regina

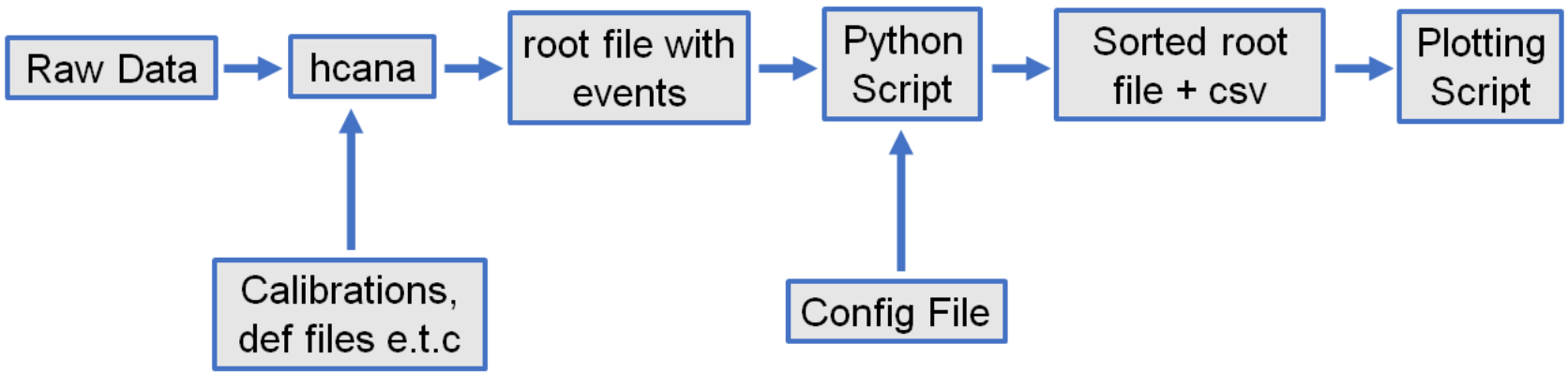
29/04/20

Introduction

- Switching to python based analysis structure
- Hope is that this will be clearer and more transportable
  - Also potentially more accessible for new group members
- Good progress being made by Richard and myself, things are starting to come together into a framework
- I will outline the rough sketch of what is planned (as I see it) in the next few slides

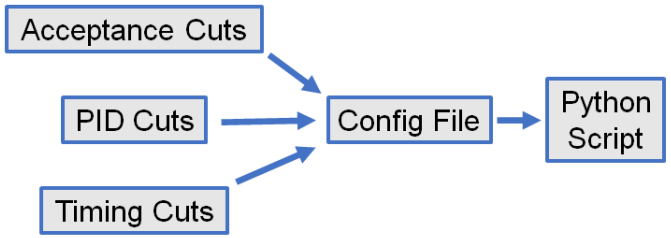
General Data Flow

- Starting from raw data, process through hcana
- Get a resulting root file based on our defined def files etc.
- Run large root file through python analysis script, get a trimmed and sorted root file and csv as output
  - Choice from the user as to which they use after that
  - Could use python based plotting/fitting if they want



Python Script Input

- Execute python script in a similar manner to old scripts, give it run number etc.
- Also provide a path to a "config" file
- Config points to a list of .csv database files
- Database files have cut values sorted by run, subdivided into three files
  - Timing cut parameters
  - Acceptance cut parameters
  - PID cut parameters
- Creates a cut "dictionary" based upon values that it reads in



Python Script Input - Database Format

- Simple CSV format, anything after a hash (#) is treated as a comment
- May add a "type" flag to the files too, denote if it is a singles/coin run for example

	A	B	C	D	E	F	G	H	I	J	K
1	Run_Start	Run_End	Bunch_Spacing	CT_Offset	Pion_Prompt_Peak	Kaon_Prompt_Peak	Proton_Prompt_Peak	RF_Offset			
2	4865	4870	4	0.25	47.25	47.1	47.8	-1 # Q2 = 3	W = 2.32	right	high eps
3	4871	4880	4	0.25	44.55	44.1	44.85	-1 # Q2 = 3	W = 2.32	centre	high eps
4	4881	4890	4	0.25	44.55	44.1	44.85	-1 # Q2 = 3	W = 2.32	left	high eps
5	4891	4891	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	right	high eps
6	4892	4912	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	left	high eps
7	4913	4924	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	centre	high eps
8	4925	4944	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	right	high eps
9	4945	4947	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	centre	high eps
10	4948	4959	4	0.25	44.65	44.65	45.15	-1 # Q2 = 2.115	W = 2.95	right	high eps
11	4965	4980	4	0.25	44.4	44.6	44.7	-1 # Q2 = 4.4	W =2.74	right	high eps

Python Script Input - Database Interpretation Example

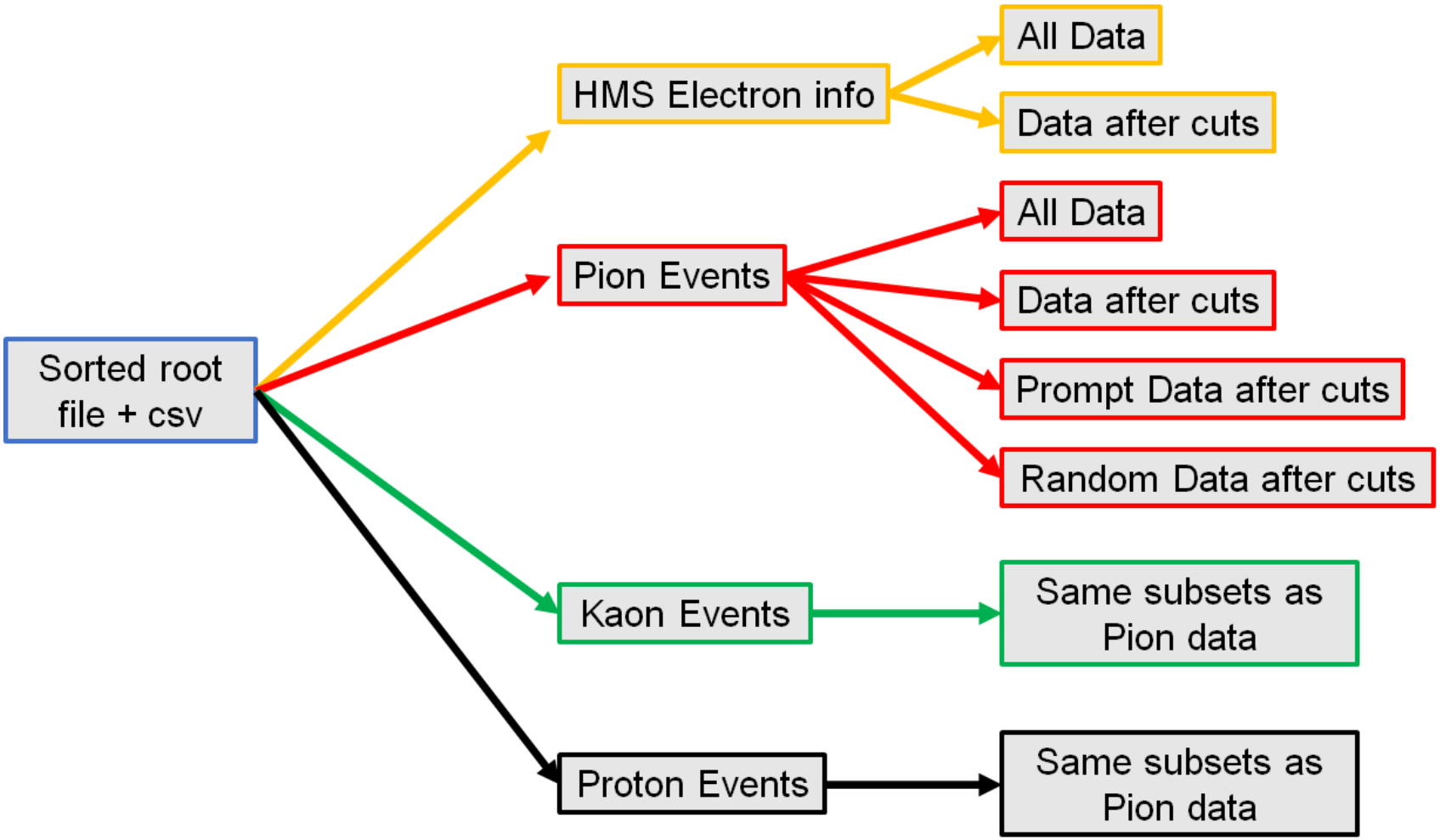
- Read in config file, get paths for parameter files
  - Complain if file is not specified!
- Loop line by line over parameter file, grab relevant info

```
Configf = open(ConfigFile)
for line in Configf:
    line = line.rstrip()
    array = line.split("= ")
    if ("Timing_Cuts" in array[0]):
        TimingTmp = array[1]
        TimingCutFile = "%s/UTIL_PROTON/config/%s" % (REPLAYPATH,TimingTmp)
    if ("PID_Cuts" in array[0]):
        PIDTmp = array[1]
        PIDCutFile = "%s/UTIL_PROTON/config/%s" % (REPLAYPATH,PIDTmp)
    if ("Acceptance_Cuts" in array[0]):
        AcceptTmp = array[1]
        AcceptCutFile = "%s/UTIL_PROTON/config/%s" % (REPLAYPATH,AcceptTmp)
Configf.close()
try:
    TimingCutFile
    PIDCutFile
    AcceptCutFile
except NameError:
    print("Error, one (or more) of the cut files is not defined in the config file provided")
    sys.exit(2)
print("Reading timing cuts from %s" % TimingCutFile)
print("Reading PID cuts from %s" % PIDCutFile)
print("Reading acceptance cuts from %s" % AcceptCutFile)
```

```
TimingCutf = open(TimingCutFile)
PromptPeak = [0, 0, 0]
linenum = 0 # Count line number we're on
TempPar = -1 # To check later
for line in TimingCutf: # Read all lines in the cut file
    linenum += 1 # Add one to line number at start of loop
    if(linenum > 1): # Skip first line
        line = line.partition('#')[0] # Treat anything after a # as a comment and ignore it
        line = line.rstrip()
        array = line.split(",") # Convert line into an array, anything after a comma is a new entry
        if(int(runNum) in range (int(array[0]), int(array[1])+1)): # Check if run number for file is within any of the ranges specified
            TempPar = 1 # If run number is in range, set to non -1 value
            BunchSpacing = float(array[2]) # Bunch spacing in ns
            CT_Offset = float(array[3]) # Offset to add to width of prompt peak
            PromptPeak[0] = float(array[4]) # Pion CT prompt peak positon
            PromptPeak[1] = float(array[5]) # Kaon CT prompt peak positon
            PromptPeak[2] = float(array[6]) # Proton CT prompt peak positon
            RF_Offset = float(array[7]) # Offset for RF timing cut
TimingCutf.close() # After scanning all lines in file, close file

if(TempPar == -1): # If value is still -1, run number provided din't match any ranges specified so exit
    print("Error, run number specified does not fall within a set of runs for which cuts are defined in %s" % AcceptCutFile)
    sys.exit(3)
```

Python Script Output



Discussion

- Does this general structure make sense to people?
- Any comments or suggestions for features that would be useful?
- Is the output structure what people want?
- How should we handle the scalers?