Hands-on: basf2

Boyang Zhang
zhangboy@hawaii.edu

Belle II Summer workshop
Durham

Contents

Use basf2 on KEKCC

Basf2: Belle Analysis Software Framework 2

- C++ modules + python interface
- Setup CL tools that are common to all releases:
  `$ source /cvmfs/belle.cern.ch/tools/b2setup`
  - ➢ Create an analysis directory, prepare the build system and adds the directory to git
    `b2analysis-create directory release`
  - ➢ Create a local development directory and prepare the build system
    `b2code-create directory [release]`
  - ➢ Setup a specific release (central or local) `b2setup release`
  - ➢ Many more can be found in the documentation

- Everyday usage:
  `$ source /cvmfs/belle.cern.ch/tools/b2setup release`

- How to setup central/local release:
  https://software.belle2.org/development/sphinx/build/tools_doc/cvmfs_setup.html

Use basf2 on KEKCC

What are available software releases?

- Main releases (release-major-minor-patch, e.g. release-06-01-10)
  - ➢ Contains all basf2 libraries (37 in total as of now), see https://software.belle2.org/monitoring/development/latest
  - ➢ Can be used for everything, (data-taking, track/cluster reconstruction, MC production, and high-level analysis)

- Light releases (light-yymm-cat_breed, e.g. light-2305-korat)
  - ➢ Will be used for today's hands-on exercise
  - ➢ Contains only libraries necessary for analyses, i.e. framework, mdst, mva, analysis, skim, geometry, online_book, b2bii
  - ➢ Contains most up to date analysis bugfix and features, e.g. BDT/NN based PID, Charm flavour tagger
  - ➢ Designed for high-level analysis only (i.e. can only process mdst/udst)

- Release history:
  What's New — basf2 development documentation

Use basf2 on KEKCC

Many CL tools are available **after** setting up a release

- The most important: to process analysis steering file
  `basf2 steering.py`
  - ➢ `--dry-run` checks the syntax but doesn't start processing
  - ➢ `-n 100` limits the event loop to 100 events
  - ➢ `-i myinputfile` provides input data file
  - ➢ `--help` prints full list of possible arguments

- Tools for file handling, e.g. `b2file-metadata-show`
- Tools for luminosity, e.g. `b2info-luminosity`
- Conditions DB interface
- And more

- Basf2 Command Line Tools:
  https://software.belle2.org/development/sphinx/framework/doc/index-02-commandline-tools.html

Contents

Basic reconstruction steering file

- What is a steering file?
  In Basf2, data events are processed one by one through a path of relevant C++ modules (wrapped in Python) executed sequentially. The python file containing the path is called the steering file.

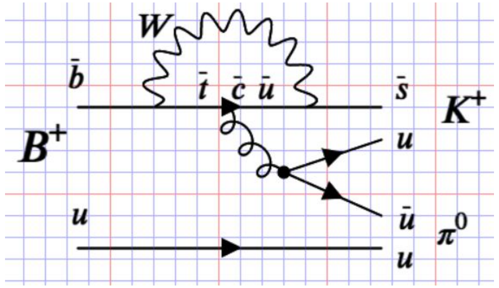
- Examples of basic elements in a steering file
  - import basf2 and modularAnalysis (optionally with short names)
  - create path, e.g. main_path = basf2.Path()
  - read input mdst or udst data using inputMdst / inputMdstList
  - create lists of final-state particles using fillParticleList
  - form composite particles using reconstructDecay
  - save variables in output ntuple using variablesToNtuple
  - process the path

- Comprehensive tutorials (highly recommended):
  Beginners' tutorials — basf2 development documentation

Contents

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

- Why $B^{\pm} \to K^{\pm} \pi^0$?
  - Flavor changing neutral current (FCNC), which is sensitive to new physics
  - One of the modes in charmless hadronic B CP-violating charge asymmetry measurements
  - Contains both charged and neutral final state particles, good practice
  - π^0 is reconstructed from 2 photons
  - Relatively clean and reconstruction is not too complicated



- Goal of this exercise:
  - Setup basf2 with light-2305-korat
  - Use signal MC on KEKCC
    /home/belle/zhangboy/Summer_workshop_2023/mdst/MC15ri_b_1210012000_1_sig.root
  - Write a steering file that:
    - Reconstruct $B^{\pm}$ meson through $B^{\pm} \to K^{\pm} \pi^0$
    - Suppress various kinds of background
    - Save continuum suppression variables to the output Ntuple
    - Save a fitting variable to the Ntuple for yield extraction

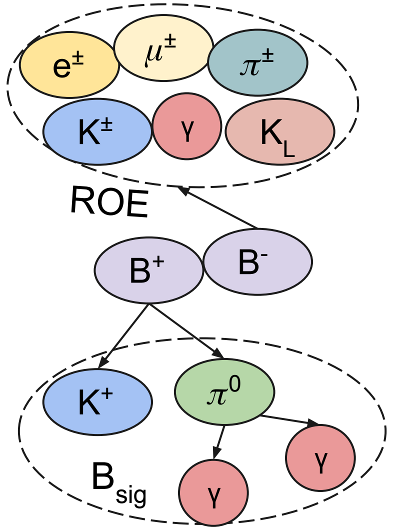
    The output Ntuple will be used for Continuum Suppression and pyhf hands-on sessions

Hands-on exercise with $B^\pm \to K^\pm \pi^0$

- What is reconstruction in analysis? (Not track/cluster reconstruction)
  In a mdst file, each event contains tracks and clusters. We select 1 track and 2 clusters to be our $K^\pm$ and π^0, combine their 4 momenta to form the $B^\pm$.

  In this process, all possible combinations are saved if there are more tracks and clusters. Thus, we will likely have multiple $B^\pm$ candidates per event.

- What events/candidates will be reconstructed besides signal?
  - From final state level, events where:
    - A clone/fake track is used for Kaon
    - A beam background or fake photon is used to form the π^0
    - 2 real photons that originate from different B mesons
  - From reconstructed level, events where:
    - A real track with wrong mass hypothesis (misID, e.g. $B^\pm \to \pi^\pm \pi^0$)
    - B decay to a final state that is $K^\pm \pi^0$ … (e.g. $B^0 \to K^\pm$ [D -> K $2\pi^\pm\pi^0$])
    - $K^\pm$ and π^0 originate from different B mesons
    - $K^\pm$ and π^0 originate from qqbar events (uubar, ddbar, ssbar, ccbar)
  - Q: How to use MC truth variables to separate them and what variables can suppress them?



Hands-on exercise with $B^{\pm}$ -> $K^{\pm}$ π^{0}

1. Import necessary libraries

```
import basf2 as b2
import modularAnalysis as ma
from variables import variables as vm
import variables.collections as vc
import variables.utils as vu
import vertex as vx
import stdPi0s
```

2. Prepend/Append global tags

```
b2.conditions.prepend_globaltag(ma.getAnalysisGlobaltag())
```

Q: What is a globaltag and globaltag replay?
Do we always need it in an analysis?
5.4. Conditions Database — basf2 development documentation
Where can I find the relevant global tag names?
https://confluence.desy.de/display/BI/Conference+readiness
https://cdbweb.sdcc.bnl.gov/GlobalTag

Hands-on exercise with $B^\pm \to K^\pm \pi^0$

3. Define a path and input file

```
main_path = b2.Path()
file_path = /home/belle/zhangboy/Summer_workshop_2023/mdst/MC15ri_b_1210012000_1_sig.root
ma.inputMdst(filelist= file_path, path=main_path)
```

Q: When modules are added to a path, they are not executed right away. What do we need to execute them?

What are paths and modules?
https://software.belle2.org/development/sphinx/framework/doc/modules_paths.html

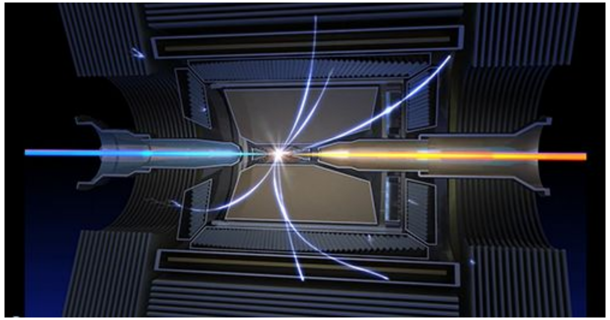
7.2.1. inputMdstList — basf2 development documentation

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

4.1 Define good tracks (high momentum)

```
goodTrack = '[abs(dz)<2] and [dr<0.5] and thetaInCDCAcceptance and [nCDCHits>20]'
```

Good impact parameters will make sure the track is originated from IP
thetaInCDCAcceptance will exclude tracks in the direction of beam pipe
The more CDCHits, the better momentum resolution
All above contribute to a precise momentum measurement



Variable definitions:
[7.3. Variables — basf2 development documentation]

Q: What is the range of CDC acceptance theta angle in degrees?
For muon tracks, inKLMAcceptance is often used in addition to the CDCAcceptance..
What is the difference? Could you justify it?

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

4.2 Fill final state particle lists - Kaon candidates

```
ma.fillParticleList(decayString='K+:myk', cut='', path=main_path)
```

Some list names are reserved and the corresponding cut cannot be modified:
all: no cut; MC: generated particle, MCparticle

Q1. How many lists do we need for $K^{\pm}$?
Q2. The $K^{\pm}$ candidate should be a good track and have a high probability being a real Kaon. What variable should we use? This will suppress misID background.

Final state particles at Belle II:
[7.2.1. fillParticleList — basf2 development documentation]

Q: How many types of final state particles are there in Belle II?
What are their mdst sources?

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

4.2 Fill final state particle lists - Kaon candidates

```
ma.fillParticleList('K+:myk', cut=goodTrack + ' and [kaonID_noSVD>0.9]', path=main_path)
```

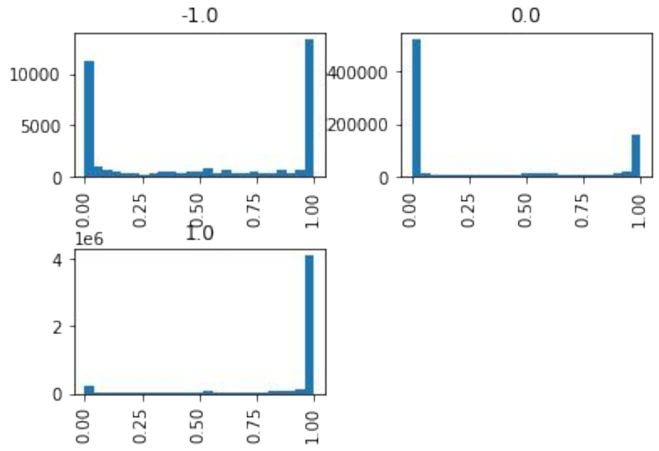
KaonID_noSVD is a likelihood based global PID.
Q: What other types PID do we have?:
7.3. PID Variables — basf2 development documentation

Plot the KaonID_noSVD:
To plot True and False Kaon, MC truth information (isSignal) is used. The output of isSignal could be:

1	if the K candidate is a real Kaon;
0	if the K candidate is a real track but not a Kaon or a clone
Nan	if the K candidate has no related MCparticle, i.e. fake track

(in the plot, Nan is rewritten as -1)

KaonID_noSVD



noSVD: dE/dx is not calibrated for this release

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

5.1 Define good ECL clusters (photon candidates)

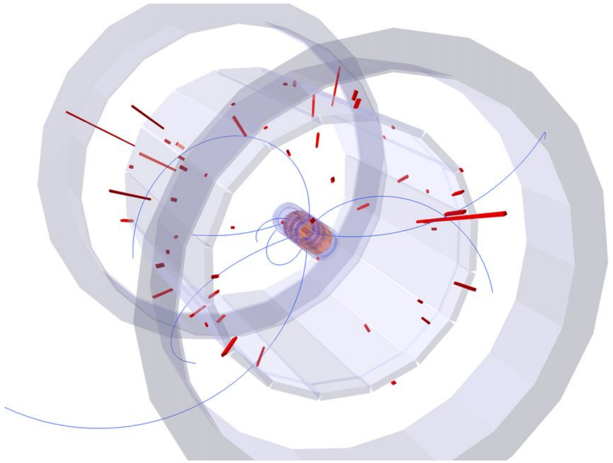
Goal: suppress beam background and fake photons

Q: What variables are good at suppressing both and what for only beam background photon?

ECL: ~8000 crystals measure the energy deposition
Hint: There are only a few high energy photons in an event, but many low energy photons. Photons whose energy is below certain threshold is mainly from beam background and electronic noise (fake). Note that this threshold depends on the ECL region.

Q: Will a photon deposit all its energy to the ECL?

Selection recommendation:
https://confluence.desy.de/display/BI/May2020+pi0+Recommendations
https://confluence.desy.de/display/BI/Neutrals+Performance

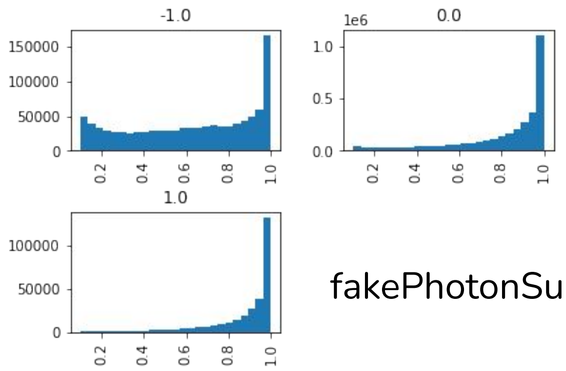

Cunliffe, S. (2017). Prospects for rare B decays at Belle II.

Hands-on exercise with B$^{\pm}$ -> K$^{\pm}$ π^0

5.1 Define good ECL clusters (photon candidates)

```
vm.addAlias('goodphoton','passesCut(
[abs(clusterTiming)<formula(2*clusterErrorTiming)] and \
[abs(clusterTiming)<200] and [clusterE>0.05] and inECLAcceptance and \
[beamBackgroundSuppression>0.2] and [fakePhotonSuppression>0.2] and
[clusterNHits>1.5])')
```

Selection recommendation:
https://confluence.desy.de/display/BI/Neutrals+Performance



fakePhotonSuppression

Hands-on exercise with $B^{\pm}$ -> $K^{\pm}\ \pi^0$

5.2 Fill standard π^0 with goodphoton daughters

```
stdPi0s.stdPi0s(listtype=,beamBackgroundMVAWeight="MC15ri",
fakePhotonMVAWeight="MC15ri",path=main_path)
```

Recommended selection and systematics are/will be wrapped into standard particles:

Q: What standard particles are ready to use?
What particles don’t have a standard particle list or not recommended.
https://software.belle2.org/development/sphinx/analysis/doc/StandardParticles.html

Hands-on exercise with $B^{\pm}$ -> $K^{\pm}$ π^0

5.2 Fill standard π^0 with goodphoton daughters

```
stdPi0s.stdPi0s(listtype='eff60_May2020',beamBackgroundMVAWeight="MC15ri",
fakePhotonMVAWeight="MC15ri",path=main_path)
```

```
ma.cutAndCopyLists(outputListName='pi0:mypi0', inputListNames='pi0:eff60_May2020',
    cut='[daughter(0,goodphoton)] and [daughter(1,goodphoton)]', path=main_path)
```

Q: What selection does eff60_May2020 apply to the π^0?
What are other options for the listtype? What are their differences?
https://confluence.desy.de/display/BI/May2020+pi0+Recommendations

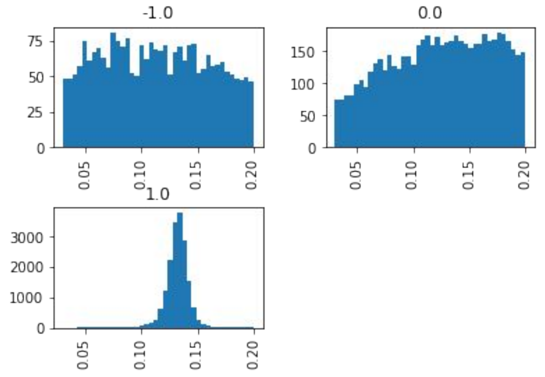
Hands-on exercise with $B^{\pm}$ -> $K^{\pm}\ \pi^0$

5.2 Fill standard π^0 with goodphoton daughters

```
        stdPi0s.stdPi0s(listtype='eff60_May2020',beamBackgroundMVAWeight="MC15ri",
fakePhotonMVAWeight="MC15ri",path=main_path)

        ma.cutAndCopyLists(outputListName='pi0:mypi0', inputListNames='pi0:eff60_May2020',
                cut='[0.115<M<0.15] and [daughter(0,goodphoton)] and [daughter(1,goodphoton)]',
path=main_path)
```

Q: What if the 2 photon daughters come from different particles?

π^0 mass



Hands-on exercise with $B^{\pm}$ -> $K^{\pm}$ π^0

6. Best candidate selection

```
ma.rankByLowest("pi0:mypi0", variable="abs(SigM)", numBest=1, path=main_path)
```

Q: What variable would you like to use for the B meson best candidate selection?

[7.2.1. rankByLowest — basf2 development documentation]()

Hands-on exercise with $B^\pm$ -> $K^\pm \pi^0$

7. Reconstruct B meson

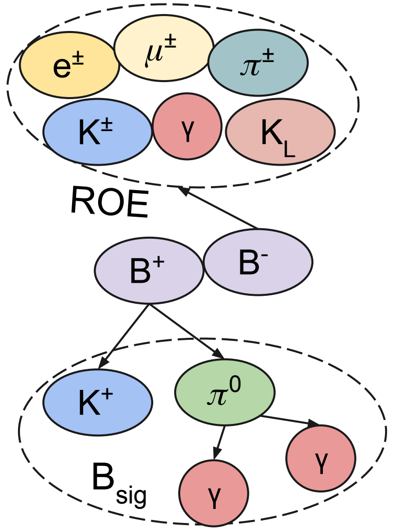
```
ma.reconstructDecay(decayString='B+:Kpi0 -> K+:myk pi0:mypi0', cut='',
path=main_path)
```

Q: Why are there different types of arrows?
Will using different arrows change the reconstruction algorithm? Or it changes the truth-matching criteria?
What are other decayString grammar?

In what case, would dmID be useful?

7.2.1. reconstructDecay — basf2 development documentation

Decay string grammar
7.1.2. DecayString — basf2 development documentation



Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

7. Reconstruct B meson

```
    ma.reconstructDecay(decayString='B+:Kpi0 -> K+:myk pi0:mypi0', cut='',
path=main_path)
```

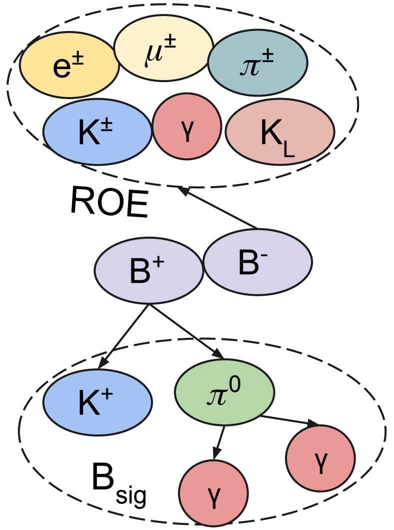
Q : Not only signal but background events are also reconstructed. How to handle?

Mbc and deltaE (The most useful variables for hadronic B decays)

$m_B = \sqrt{(E^2_B - p^2_B)}$, $E_{beam} = \sqrt{s} / 2$, well measured beam energy $\sqrt{s}/2$

$M_{bc} = \sqrt{(E^2_{beam} - p^{*2}_B)}$, $\Delta E = E^*_B - E_{beam}$

Note that V* is a variable measured in CMS frame



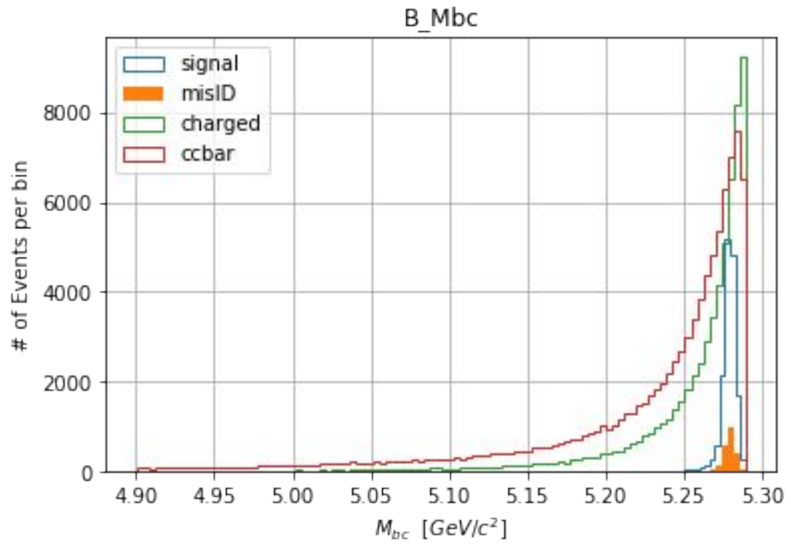
Hands-on exercise with B$^\pm$ -> K$^\pm$ π^0

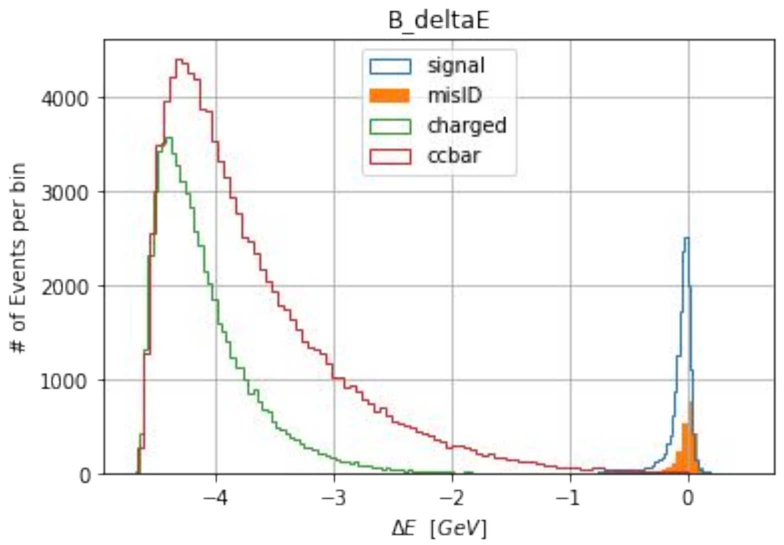
7. Reconstruct B meson

```
    ma.reconstructDecay(decayString='B+:Kpi0 -> K+:myk pi0:mypi0', cut='Mbc>5.25',
path=main_path)
```

$M_{bc} = \sqrt{(E^2_{beam} - p^{*2}_B)}$

$\Delta E = E^*_B - E_{beam}$





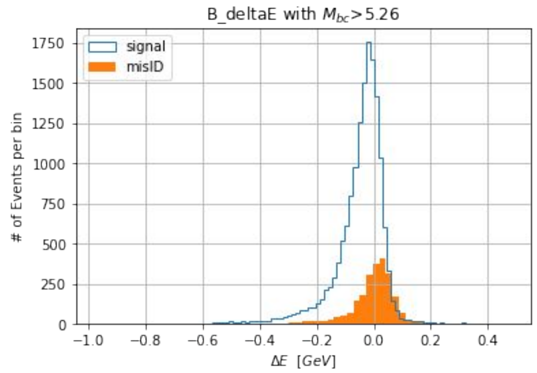
Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

7. Signal region for M_{bc} and ΔE

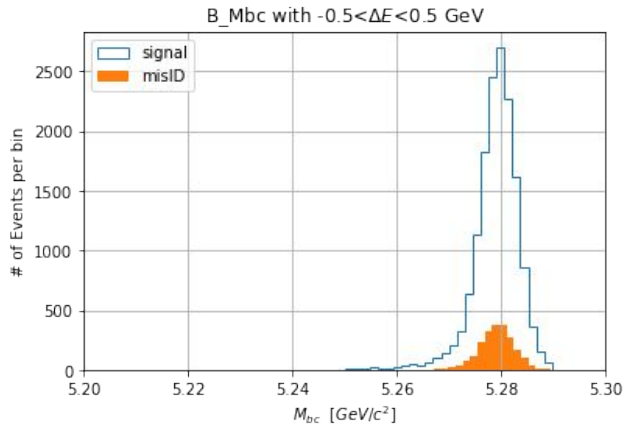
$M_{bc} = \sqrt{(E^2_{beam} - p^{*2}_B)}$ $\Delta E = E^*_B - E_{beam}$

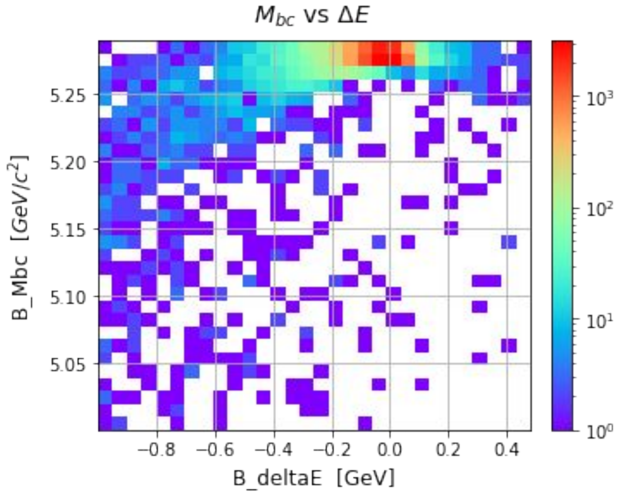
$E_{beam} = \sqrt{s} / 2$

Note that BBbar and qqbar are plotted in the 2d hist but not in the projections.



Q: Why does signal overlap with misID in M_{bc} but not in ΔE?





Which one is a better candidate as a fitting variable for this mode!

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

8. Truth matching (only possible for MC, not data)

```
ma.matchMCTruth('B+:Kpi0', path=main_path)
```

Setup the relation between the reconstructed particles and generated particles

https://software.belle2.org/development/sphinx/analysis/doc/MCMatching.html

7.3. Truth Variables — basf2 development documentation

It is always useful to save various truth variables to the output Ntuple. E.g:

isSignal - Yes/No if the reconstructed particle matches the generated particle

Hands-on exercise with $B^{\pm}$ -> $K^{\pm}\ \pi^0$

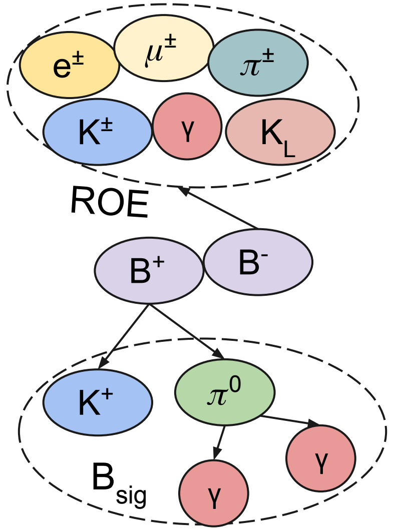
8. What information/variables do we need?

Signal events - `B_isSignal==1`

misID events - If the B meson and daughter relations are correct but only Kaon mass hypothesis is wrong. (We can check the PDG code for each reconstructed particle, but...)

BBbar background - If the reconstructed B meson is a real B (charged or neutral), and (If K π^0 have different B mother, or if any final state particle is missing). Again, it is possible to know this by checking the PDG code

qqbar background - If K π^0 decay from a qqbar event.



Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

8. Truth matching variables

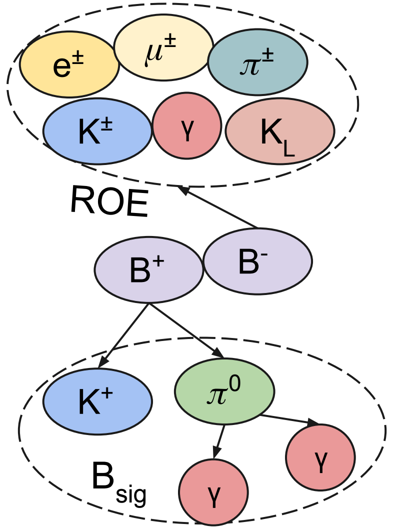
mcPDG - the PDG of the generated particle that is matched to the reconstructed particle (For a composite particle, mcPDG will return the PDG of the common generated mother of reconstructed daughter particles)

Q: In what case, would you see B_mcPDG == 300553?

mcErrors - the error flag of the reconstructed particle
https://software.belle2.org/development/sphinx/analysis/doc/MCMatching.html#error-flags

genMotherPDG - the PDG of the generated mother of the reconstructed particle

isContinuumEvent - if the generated event is a continuum event



Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

8. What information/variables do we need?

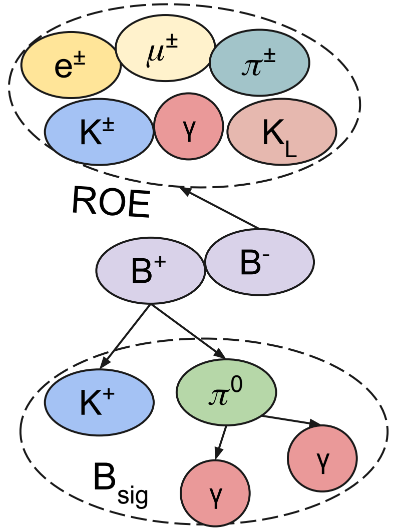
Signal events - B_isSignal==1

misID events - B_mcErrors==128

BBbar background - B_mcErrors>0 and B_mcErrors!=128 and B_isContinuumEvent==0

qqbar background - B_isContinuumEvent==1

Q: Could you verify the selections?
Could you achieve the same classification without using B_mcErrors?



Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

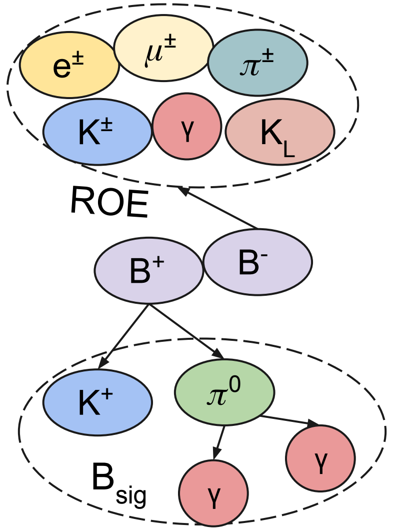
9. Continuum suppression and Rest of event

```
ma.buildContinuumSuppression(list_name="B+:Kpi0",
roe_mask="my_mask", path=main_path)
```

Tutorial:
[3.4.10. Continuum Suppression (CS) — basf2]

What is the **roe_mask**?

Continuum suppression variables are event shape variables, their calculations involve technically all tracks and clusters in an event, i.e. both B_{sig} and B_{tag}. The collection of tracks and clusters that are not used to form the B_{sig} is called Rest Of Event (ROE). However, ROE contains not only true tracks/clusters, there are also clone/fake/bkg and poorly measured tracks/clusters. Analysts usually apply a mask (set of cuts) on ROE to exclude possible clones/fakes/bkg.

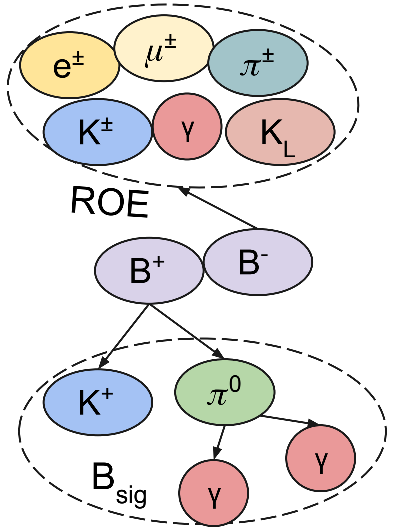


Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

9. Rest of event

```
ma.fillParticleList('gamma:all', '', path=main_path)
ma.getBeamBackgroundProbability('gamma:all','MC15ri', path=main_path)
ma.getFakePhotonProbability('gamma:all','MC15ri', path=main_path)
ma.buildRestOfEvent('B+:Kpi0', fillWithMostLikely=True, path=main_path)

good_track = '[abs(dz)<20] and [dr<10] and thetaInCDCAcceptance and [nCDCHits>0] and [pt>0.075]'
good_gamma = f'[clusterE>0.05] and [clusterNHits>1.5] and [abs(clusterTiming)<formula(2*clusterErrorTiming)] and [abs(clusterTiming)<200] and [beamBackgroundSuppression>0.2] and [fakePhotonSuppression>0.2]'
roe_mask = ('my_mask', good_track, good_gamma)
ma.appendROEMasks('B+:Kpi0', [roe_mask], path=main_path)

# creates V0 particle lists and uses V0 momentum to update the Rest Of Event
ma.updateROEUsingV0Lists('B+:Kpi0', mask_names='my_mask', default_cleanup=True, selection_cuts=None, apply_mass_fit=True, fitter='treefit', path=main_path)
```

Tutorial: https://software.belle2.org/development/sphinx/analysis/doc/RestOfEvent.html



Hands-on exercise with B$^{\pm}$ -> K$^{\pm}$ π^0

10. Event shape variables

```
ma.buildEventShape(cleoCones=False, collisionAxis=False, foxWolfram=True,
                   harmonicMoments=False, jets=False, sphericity=False, path=main_path)
ESVariables = ['foxWolframR1', 'foxWolframR4', 'thrustAxisCosTheta']
# there are more event shape variables, but here only 3 are used
```

Documentation:
[7.2.1. buildEventShape — basf2 development documentation]

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

11. Tag vertex and vertex fitting

```
# signal-side B vertex fit is required for the tagV
vx.treeFit('B+:Kpi0', conf_level=0.00, updateAllDaughters=False, massConstraint=[ ], ipConstraint=True, path=main_path)

# fit the tag-side B vertex
vx.TagV('B+:Kpi0',confidenceLevel=0.0,trackFindingType='standard_PXD',MCassociation='breco',constraintType='tube', reqPXDHits=0, maskName='my_mask', fitAlgorithm='KFit', kFitReqReducedChi2=5, path=main_path)
vm.addAlias('TagVReChi2','formula(TagVChi2/TagVNDF)')
vm.addAlias('TagVReChi2IP','formula(TagVChi2IP/TagVNDF)')
```

Vertex fitting (Treefit, Kfit and TagV):
https://software.belle2.org/development/sphinx/analysis/doc/VertexFitting.html

Hands-on exercise with $B^{\pm}$ -> $K^{\pm}$ π^0

10. Variables to be used in the Continuum Suppression hands-on

```
ESVariables = ['foxWolframR1',           'foxWolframR4',       'thrustAxisCosTheta']

TVVariables = ['DeltaZ',    'DeltaZErr',    'TagVReChi2',    'TagVReChi2IP',
               'TagVx',     'TagVxErr',     'TagVy',         'TagVyErr',
               'TagVz',     'TagVzErr',     'TagVNTracks']

CSVariables = ['isContinuumEvent',      "R2",                     "thrustBm",
               "thrustOm",               "cosTBTO",                "cosTBz",
               'KSFWV_et',               'KSFWV_mm2',              'KSFWV_hso00',
               'KSFWV_hso01',            'KSFWV_hso02',            'KSFWV_hso03',
               'KSFWV_hso04',            'KSFWV_hso10',            'KSFWV_hso12',
               'KSFWV_hso14',            'KSFWV_hso20',            'KSFWV_hso22',
               'KSFWV_hso24',            'KSFWV_hoo0',             'KSFWV_hoo1',
               'KSFWV_hoo2',             'KSFWV_hoo3',             'KSFWV_hoo4',
               "CC1",                    "CC2",                    "CC3",
               "CC4",                    "CC5",                    "CC6",
               "CC7",                    "CC8",                    "CC9", ]
```

Hands-on exercise with B$^\pm$ -> K$^\pm$ π^0

10. Variables to be used in the Continuum Suppression hands-on

```
vm.addAlias('KSFWV_et','KSFWVariables(et)')
vm.addAlias('KSFWV_mm2','KSFWVariables(mm2)')
vm.addAlias('KSFWV_hso00','KSFWVariables(hso00)')
vm.addAlias('KSFWV_hso01','KSFWVariables(hso01)')
vm.addAlias('KSFWV_hso02','KSFWVariables(hso02)')
vm.addAlias('KSFWV_hso03','KSFWVariables(hso03)')
vm.addAlias('KSFWV_hso04','KSFWVariables(hso04)')
vm.addAlias('KSFWV_hso10','KSFWVariables(hso10)')
vm.addAlias('KSFWV_hso12','KSFWVariables(hso12)')
vm.addAlias('KSFWV_hso14','KSFWVariables(hso14)')
vm.addAlias('KSFWV_hso20','KSFWVariables(hso20)')
vm.addAlias('KSFWV_hso22','KSFWVariables(hso22)')
vm.addAlias('KSFWV_hso24','KSFWVariables(hso24)')
vm.addAlias('KSFWV_hoo0','KSFWVariables(hoo0)')
vm.addAlias('KSFWV_hoo1','KSFWVariables(hoo1)')
vm.addAlias('KSFWV_hoo2','KSFWVariables(hoo2)')
vm.addAlias('KSFWV_hoo3','KSFWVariables(hoo3)')
vm.addAlias('KSFWV_hoo4','KSFWVariables(hoo4)')
vm.addAlias('CC1','CleoConeCS(1)')
vm.addAlias('CC2','CleoConeCS(2)')
vm.addAlias('CC3','CleoConeCS(3)')
vm.addAlias('CC4','CleoConeCS(4)')
vm.addAlias('CC5','CleoConeCS(5)')
vm.addAlias('CC6','CleoConeCS(6)')
vm.addAlias('CC7','CleoConeCS(7)')
vm.addAlias('CC8','CleoConeCS(8)')
vm.addAlias('CC9','CleoConeCS(9)')
```

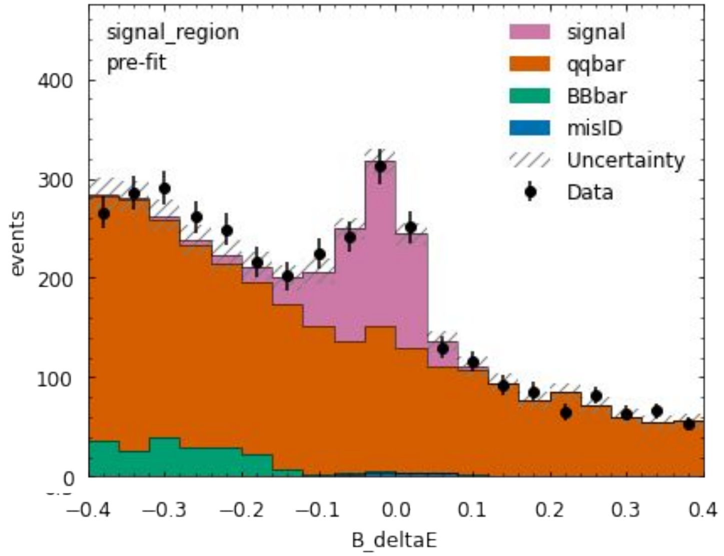
Hands-on exercise with B± -> K± π^0

12. Save variables to the output Ntuple

```
vm.addAlias('CMS_cosTheta', 'useCMSFrame(cosTheta)')

b_vars = vu.create_aliases_for_selected(
    list_of_variables= vc.deltae_mbc + CSVariables + ESVariables +
TVVariables + ['mcErrors','mcPDG','isSignal','CMS_cosTheta','M'],
    decay_string='^B+:Kpi0 -> K+:myk pi0:mypi0',
    prefix=['B'])

k_vars = vu.create_aliases_for_selected(
 list_of_variables=['binaryKID','kaonID_noSVD','pionID_noSVD',
'kaonIDNN','pionIDNN', 'genMotherPDG','mcPDG','isSignal'],
    decay_string='B+:Kpi0 -> ^K+:myk pi0:mypi0',
    prefix=['K'])

pi0_vars = vu.create_aliases_for_selected(
    list_of_variables=
    ['genMotherPDG','mcErrors', 'mcPDG','isSignal','M'],
    decay_string='B+:Kpi0 -> K+:myk ^pi0:mypi0',
    prefix=['pi'])
```

[7.3. create_aliases_for_selected — basf2]

B± -> K± π^0



With R2<0.45 and cosTBTO<0.8
Full CS will give better sig/bkg ratio

Hands-on exercise with B± -> K± π^0

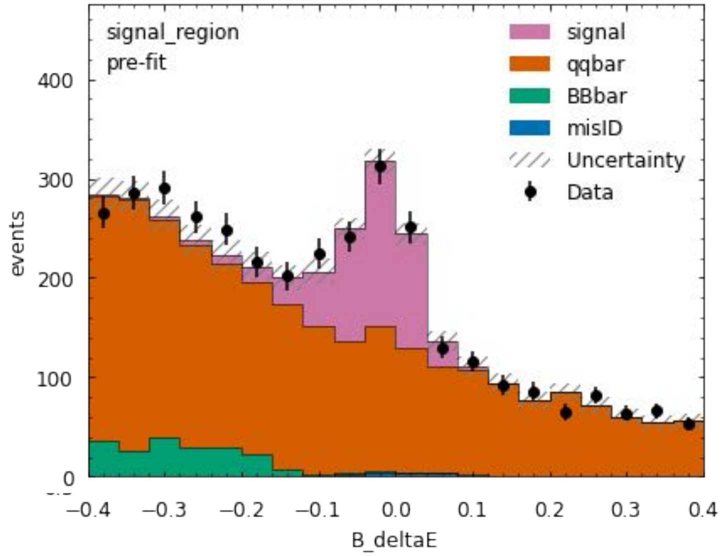
12. Save variables to the output Ntuple

```
    gamma_vars = vu.create_aliases_for_selected(
    list_of_variables=['genMotherPDG','mcPDG','isSignal',
'beamBackgroundSuppression', 'fakePhotonSuppression'],
        decay_string='B+:Kpi0 -> K+:myk [pi0:mypi0 ->
^gamma:eff60_May2020 ^gamma:eff60_May2020]',
        prefix=['g1','g2'])

    candidate_vars = ['Ecms'] + b_vars + k_vars + pi0_vars +
gamma_vars

    ma.variablesToNtuple('B+:Kpi0', candidate_vars,
basketsize=20000000, filename=output_file, treename='Bsig',
path=main_path)
```

[7.2.1. variablesToNtuple — basf2]

Load Ntuples to pandas dataframe
[3.3.3. Python — basf2 development documentation]

B± -> K± π^0



With R2<0.45 and cosTBTO<0.8
Full CS will give better sig/bkg ratio

Hands-on exercise with $B^{\pm} \to K^{\pm} \pi^0$

13. **Process the path**

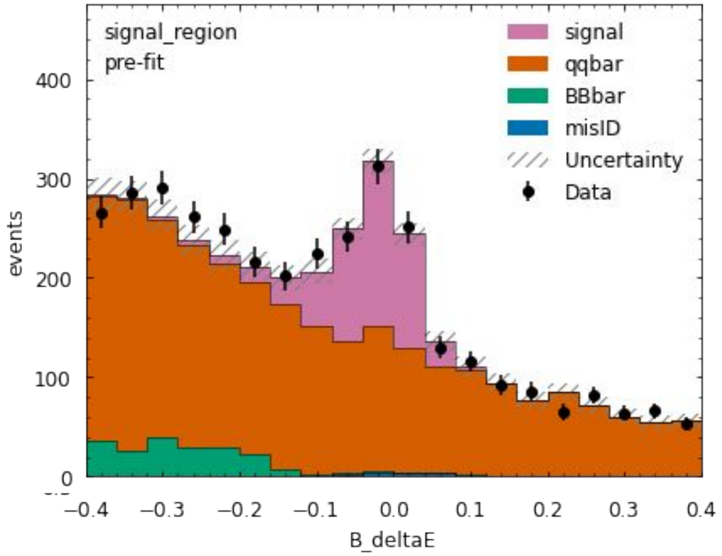
```
b2.process(path=main_path)
print(b2.statistics)
```

Documentation:

https://software.belle2.org/development/sphinx/framework/doc/modules_paths.html

https://software.belle2.org/development/sphinx/framework/doc/module_statistics.html

$B^{\pm} \to K^{\pm} \pi^0$



With R2<0.45 and cosTBTO<0.8
Full CS will give better sig/bkg ratio

Contents

Summary

If you take your analysis as an exam (100 pt total) , you will be guaranteed:

1. 60 pt if you read and understand: Sphinx documentation (excellently written)
   basf2 development documentation

2. 10 pt if you read and understand: Example scripts
   https://gitlab.desy.de/belle2/software/basf2/-/tree/main/analysis/examples

3. 10 pt if you subscribe to Questions forum
   Belle 2 Questions

4. 20 pt if you work hard, communicate well with your advisor and other students

Mahalo!

Backup:

Frank’s slides and examples from last year’s basf2 hands-on

https://indico.belle2.org/event/6444/contributions/37572/