BCO Calibration module

Jaein Hwang

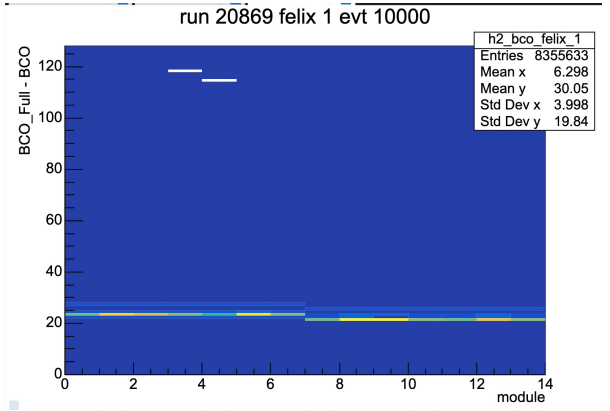
Overview diagram to create/load BCO info

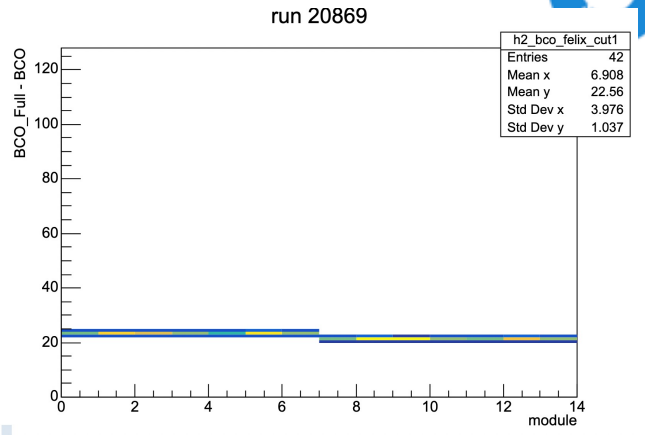
https://github.com/gwd213/INTT/tree/main/general_codes/Jaein/BCOFinder

BCOFinder.cc
-Making BCO distribution(TH2D)
- Finding peak position

ROOT File (TH2D)

CDBTTree





Update

InttBCOMap.cc/h

- Importing CDBTTree
- Masking BCO background

https://github.com/sPHENIX-Collaboration/INTT/tree/main/general_codes/Jaein/Calibration

```
root [1] .ls
TFile**         cdb_bco_20869.root
 TFile*         cdb_bco_20869.root
  KEY: TTree    Single;1        Single
  KEY: TTree    Multiple;1      Multiple
root [2] Multiple->Print()
******************************************************************************
*Tree    :Multiple  : Multiple                                               *
*Entries :      112 : Total =            4499 bytes  File  Size =       1347 *
*       :          : Tree compression factor =   3.02                       *
******************************************************************************
*Br    0 :IID       : IID/I                                                  *
*Entries :      112 : Total  Size=        999 bytes  File Size  =        260 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   2.01     *
*.............................................................................*
*Br    1 :Ibco_diff : Ibco_diff/I                                            *
*Entries :      112 : Total  Size=       1029 bytes  File Size  =        160 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   3.30     *
*.............................................................................*
*Br    2 :Ifelix_channel : Ifelix_channel/I                                  *
*Entries :      112 : Total  Size=       1054 bytes  File Size  =        144 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   3.70     *
*.............................................................................*
*Br    3 :Ifelix_server : Ifelix_server/I                                    *
*Entries :      112 : Total  Size=       1049 bytes  File Size  =        137 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   3.88     *
*.............................................................................*
```

Including BCO peak information (half ladder by half ladder)

Structure of the code(1)

Basically, this code follows Takashi's suggested template in order to have consistent structure with other parameters (DAC value)

```
#ifndef INTTBCOMAP_H
#define INTTBCOMAP_H

#include <array>
#include <string>
#include "InttMapping.h"

class CDBTTree;

class InttBCOMap
{
public:
  InttBCOMap() = default;
  virtual ~InttBCOMap() {}

  virtual int LoadFromCDB(std::string const &calibname);
  virtual int LoadFromFile(std::string const &filename);

  virtual bool IsBad(int const &felix_server,
                     int const &fexlix_channel,
                     Long64_t const &bco_full,
                     const int &bco);

  virtual bool IsBad(InttNameSpace::RawData_s const &rawdata, Long64_
  virtual bool IsBad(InttNameSpace::Offline_s const &offline, Long64_


protected:
  int LoadFromCDBTTree(CDBTTree &cdbttree);
private:
  typedef std::array<std::array<int, 14>, 8> BCOArray;
  BCOArray m_bco; //[Felix server][Felix channel]
};
#endif
```

Two methods to load CDBTTree

From CDB database

```
int InttBCOMap::LoadFromCDB(std::string const &calibname)
{
  if (calibname.empty())
  {
    std::cout << "InttBCOMap::LoadFromCDB(std::string const& name)" << std::endl;
    std::cout << "\tArgument 'name' is empty string" << std::endl;
    return -1;
  }

  std::string database = CDBInterface::instance()->getUrl(calibname);
  if (database.empty())
  {
    std::cout << "InttBCOMap::LoadFromCDB(std::string const& name)" << std::endl;
    std::cout << "\tArgument 'database' is empty string. calibname invalid :" << calibname << std::endl;
    return -1;
  }

  return LoadFromFile(database);
}
```

From CDBTTree directly

```
int InttBCOMap::LoadFromFile(std::string const &filename)
{
  if (filename.empty())
  {
    std::cout << "InttBCOMap::LoadFromFile(std::string const& name)" << std::endl;
    std::cout << "\tArgument 'filename' is empty string" << std::endl;
    return 1;
  }

  if (!std::filesystem::exists(filename))
  {
    std::cout << "int InttBCOMap::LoadFromFile(std::string const& filename)" << std::endl;
    std::cout << "\tFile '" << filename << "' does not exist" << std::endl;
    return 1;
  }
```

Structure of the code(2)

```
#ifndef INTTBCOMAP_H
#define INTTBCOMAP_H

#include <array>
#include <string>
#include "InttMapping.h"

class CDBTTree;

class InttBCOMap
{
public:
  InttBCOMap() = default;
  virtual ~InttBCOMap() {}

  virtual int LoadFromCDB(std::string const &calibname);
  virtual int LoadFromFile(std::string const &filename);

  virtual bool IsBad(int const &felix_server,
                     int const &fexlix_channel,
                     Long64_t const &bco_full,
                     const int &bco);

  virtual bool IsBad(InttNameSpace::RawData_s const &rawdata, Long64_t const &bco_full, const int &bco);
  virtual bool IsBad(InttNameSpace::Offline_s const &offline, Long64_t const &bco_full, const int &bco);

protected:
  int LoadFromCDBTTree(CDBTTree &cdbttree);
private:
  typedef std::array<std::array<int, 14>, 8> BCOArray;
  BCOArray m_bco; //[Felix server][Felix channel]
};
#endif
```

```
bool InttBCOMap::IsBad(InttNameSpace::RawData_s const& raw, Long64_t const &bco_full, const int &bco)
{
  return IsBad(raw.felix_server,raw.felix_channel,bco_full,bco);
}

bool InttBCOMap::IsBad(InttNameSpace::Offline_s const & off, Long64_t const& bco_full, const int &bco)
{
  return IsBad(InttNameSpace::ToRawData(off),bco_full,bco);
}
```

Felix server, felix channel, bco_full, bco are required as input

If the hit does not belong to [peak-1,peak+1] (3 BCK cut),
return true;

std::array<<std::array<int, 14>, 8> BCOArray;
BCOArray m_bco[Felix server][Felix channel] -> used to import BCO peak

Testing InttBCOMap.cc/h(1)

Confirmed it thorugh making BCO distribution before/after applying BCO Filter with data

SimpleTestCode.cc also has been uploaded in INTT GitHub repo.

```
20  void SimpleTestCode()
21  {
22
23      gROOT->SetBatch(kTRUE);
24      recoConsts *rc = recoConsts::instance();
25      rc->set_StringFlag("CDB_GLOBALTAG", "jaein213");
26      rc->set_uint64Flag("TIMESTAMP", 20869);
27      for (int j = 0; j < 8; j++)
28      {
29          h2_bco_ladder[j] = new TH2D(Form("h2_bco_felix_%d", j), Form("h2_bco_felix_%d", j), 14, 0, 14, 128, 0, 128);
30          h2_bco_ladder_cut[j] = new TH2D(Form("h2_bco_felix_cut%d", j), Form("h2_bco_felix_cut%d", j), 14, 0, 14, 128, 0, 128);
31      }
32
33      InttBCOMap bcomap;
34  //    bcomap.LoadFromFile("/sphenix/tg/tg01/commissioning/INTT/QA/bco_bcofull_difference/CDB/2023/cdb_bco_20869.root");
35      bcomap.LoadFromCDB("INTT_BCOMAP");
```

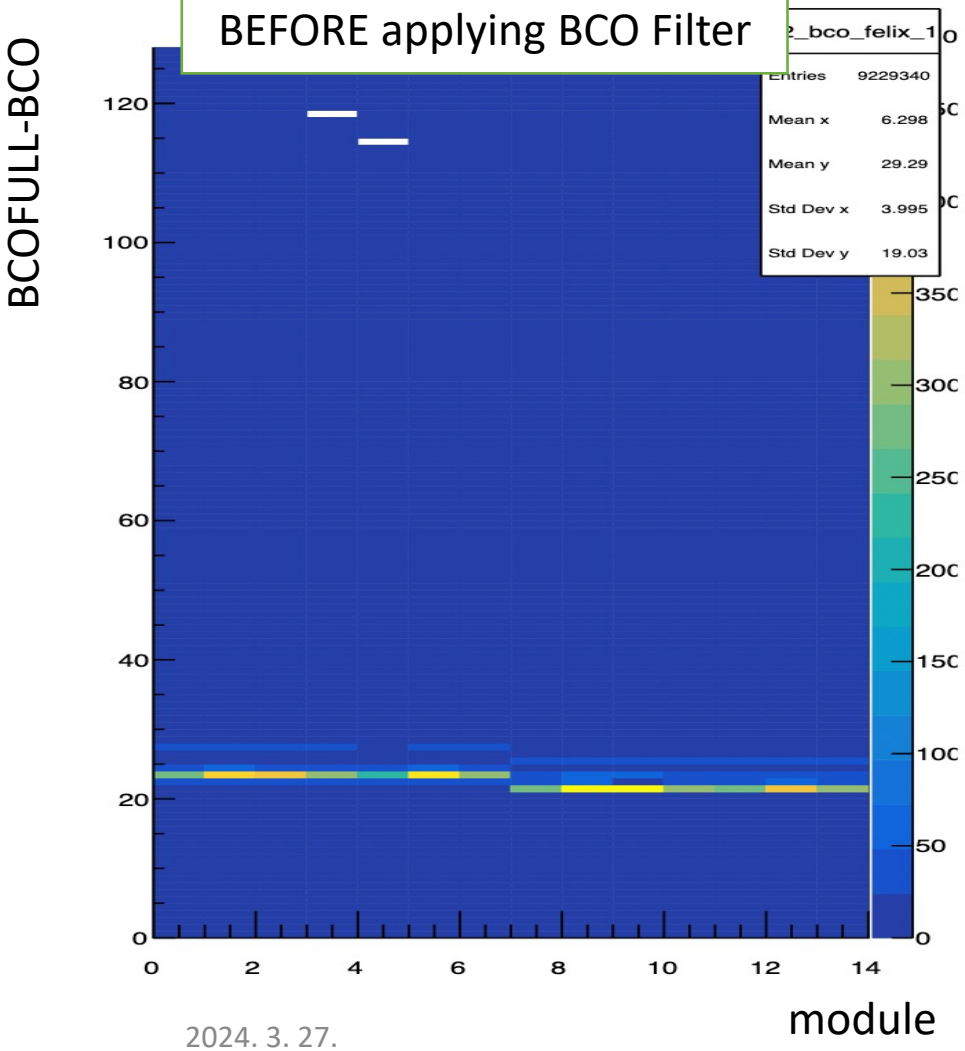
https://github.com/sPHENIX-Collaboration/INTT/blob/main/general_codes/Jaein/Calibration/SimpleTestCode/SimpleTestCode.cc

Location of original CDBTTree used to test
/sphenix/tg/tg01/commissioning/INTT/QA/bco_bcofull_difference/CDB/2023/cdb_bco_20869.root

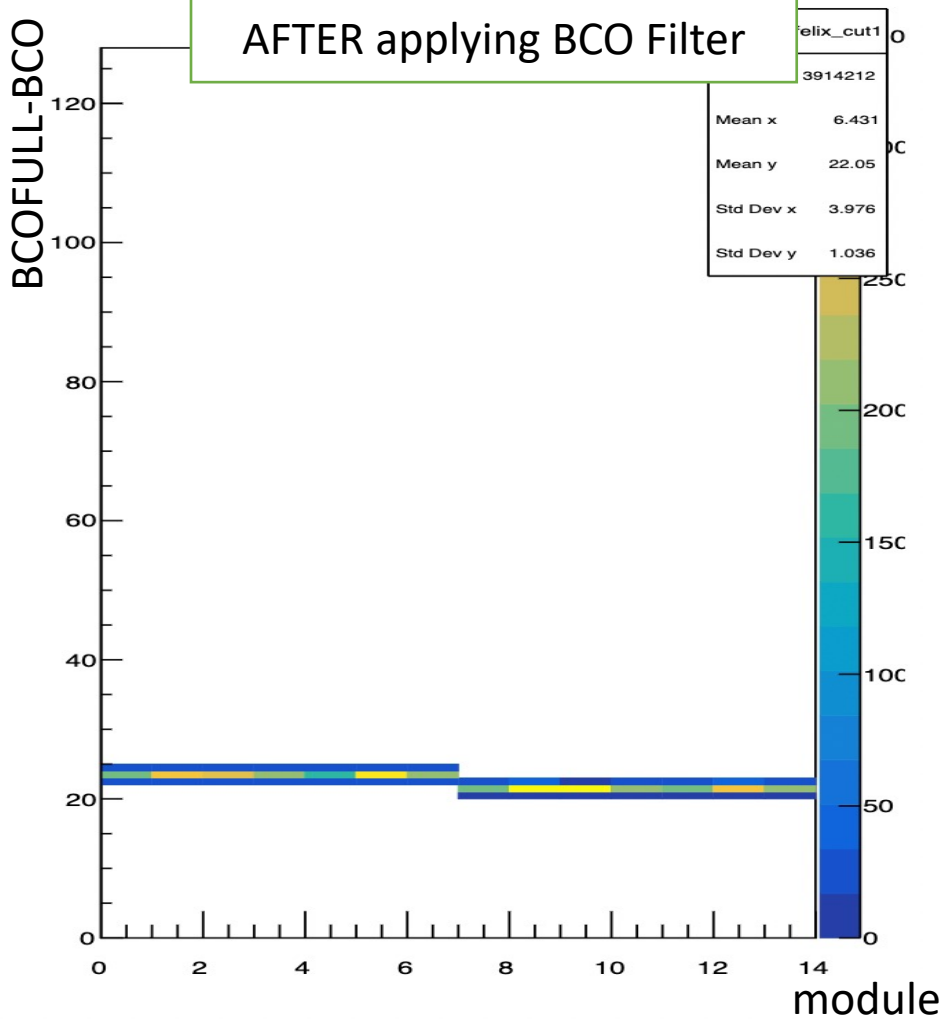
Testing InttBCOMap.cc/h(2)

https://github.com/sPHENIX-Collaboration/INTT/blob/main/general_codes/Jaein/Calibration/SimpleTestCode/SimpleTestCode.cc

BCO distribution of Felix 1
BEFORE applying BCO Filter



BCO distribution of Felix 1
AFTER applying BCO Filter



Confirmed that
1. making CDBTTree
2. Importing CDBTTree
3. Applying BCOFilter
OK

Summary

Module for BCO Filter has been published and tested.

Confirmed that inside of CDBTTree and importing/applying CDBTTree are OK

2nd version of Analysis Note for HotDeadMap and BCOFilter is available in inveno.
Yuka already made a nice analysis note for her JPS meeting.
I've added more explanations and plots and converted to sPHENIX official AN format.

https://sphenix-invenio.sdcc.bnl.gov/communities/sphenixcommunity/requests/2a116b18-dc9a-433f-a33f-8df1de29a3c9

backup

Two methods for calling CDBTTree