DC Group Meeting, 3/29/18

Timothy B. Hayward, Torri Roark

- Align the drift chambers by finding the appropriate “offsets” to minimize the residuals of straight track (B=0) data.

- run 2467 was an engineering run taken with no magnetic field: /volatile/clas12/data/rg-a/calibration/recooked/out_clas_002467*

- For now I’ve copied 10 test files to /volatile/clas12/thayward/drift_chambers/straight_track_runs/

$$\chi^2 = \Sigma_{tracks}\Sigma_{hits}\frac{(|D_{track,hit}| - |D_{hit}|)^2}{(\sigma^2_{track,hit} + \sigma^2_{hit})}$$

where

- $D_{track,hit}$ = DOCA (Distance Of Closest Approach) to the wire of the fitted track (also known as FITDOCA),
- D_{hit} = calculated DOCA from x vs t function (also known as CALCDOCA),
- $\sigma_{track,hit}$ = uncertainty in track position,
- σ_{hit} = time–based resolution of the hit.

CLAS-NOTE 2002-010, S. A. Marrow & M. D. Mestayer

- Relevant bank definitions: /coatjava/etc/bankdefs/clas12/
  - DC.xml and (?) RECEVENT.xml

```
<bank name="TimeBasedTrkg" tag="1320" info="reconstructed time-based tracking DC info">
    <section name="TBHits" tag="1321" info="reconstructed time-based tracking DC hits">
        <column name="id" num="0" type="int32" info="hit id"/>
        <column name="sector" num="1" type="int32" info="hit sector"/>
        <column name="superlayer" num="2" type="int32" info="hit superlayer"/>
        <column name="layer" num="3" type="int32" info="hit layer"/>
        <column name="wire" num="4" type="int32" info="hit wire"/>
        <column name="LR" num="5" type="int32" info="left-right ambiguity assignment"/>
        <column name="time" num="6" type="float64" info="time"/>
        <column name="doca" num="7" type="float64" info="distance to the wire"/>
        <column name="trkDoca" num="13" type="float64" info="doca to segement fit line (cm)"/>
        <column name="X" num="8" type="float64" info="hit x-coordinate  in tilted-sector"/>
        <column name="Z" num="9" type="float64" info="hit z-coordinate  in tilted-sector"/>
        <column name="clusterID" num="10" type="int32" info="associated cluster ID"/>
        <column name="timeResidual" num="14" type="float64" info="hit residual"/>
        <column name="docaError" num="15" type="float64" info="krishna: 4/1/16: Error on distance to the wire"/>
        <column name="trkID" num="16" type="int32" info="associated HB trk ID"/>
        <column name="B" num="17" type="float64" info="B-field intensity at hit location in local tilted frame"
        />
```

Possibly useful: doca, timeResidual, docaError, etc?

General Procedure

- Andrey Kim developing a method to implement shifts and rotations into the geometry package.
- Calculate χ^2 for candidate events (to be defined?)
- Implement some shift and run reconstruction on the same events.
- Recalculate χ^2.

- …

- Somehow efficiently find offsets that minimize χ^2.

```
HipoDataSource reader = new HipoDataSource();
filename = args[0]; // input file at command line
reader.open(filename);

GenericKinematicFitter fitter = new GenericKinematicFitter(11.00);

int i = 0; // counter for desired events (definition coming soon)
int n_events = 10000; //
for(int j=0; j<n_events; j++){ // limit to a certain number of events defined by n_events
// while(reader.hasEvent()==true){ // OR cycle through the entire input file
    HipoDataEvent event = reader.getNextEvent();
    boolean banks_test = true; // check to see if the event has all of the necessary banks present
    if (!(event.hasBank("REC::Particle"))) {
        banks_test = false;
        // println("no event bank"); // warning for no reconstructed event bank
    } else if (!(event.hasBank("TimeBasedTrkg::TBHits"))) {
        banks_test = false;
        // println("no time based tracking banks"); // warning for no TimeBasedTrkg bank
    }
    if (banks_test) { // check that required banks are present
        HipoDataBank eventBank = (HipoDataBank) event.getBank("REC::Particle");
        HipoDataBank hitBank = (HipoDataBank) event.getBank("TimeBasedTrkg::TBHits");
        if (eventBank.rows()==1) { // limit to one reconstructed particle
            int PID = eventBank.getInt("pid", 0); // PID, kinematic cuts to be improved later?
            if (PID==11) { // check that particle was an electron
                float fitTrack = hitBank.getFloat("trkDoca",0); // 0 = index of particle
                float residuals = hitBank.getFloat("timeResidual",0);
                float docaError = hitBank.getFloat("docaError",0);
                println("Fit track: "+fitTrack+", Residuals: "+residuals+", docaError: "+docaError);
                i++;
            }
        }
    }
}
```

```java
HipoDataSource reader = new HipoDataSource();
filename = args[0]; // input file at command line
reader.open(filename);

GenericKinematicFitter fitter = new GenericKinematicFitter(11.00);

int i = 0; // counter for desired events (definition coming soon)
int n_events = 10000; //
for(int j=0; j<n_events; j++){ // limit to a certain number of events defined by n_events
// while(reader.hasEvent()==true){ // OR cycle through the entire input file
    HipoDataEvent event = reader.getNextEvent();
    boolean banks_test = true; // check to see if the event has all of the necessary banks present
    if (!(event.hasBank("REC::Particle"))) {
        banks_test = false;
        // println("no event bank"); // warning for no reconstructed event bank
    } else if (!(event.hasBank("TimeBasedTrkg::TBHits"))) {
        banks_test = false;
        // println("no time based tracking banks"); // warning for no TimeBasedTrkg bank
    }
    if (banks_test) { // check that required banks are present
        HipoDataBank eventBank = (HipoDataBank) event.getBank("REC::Particle");
        HipoDataBank hitBank = (HipoDataBank) event.getBank("TimeBasedTrkg::TBHits");
        if (eventBank.rows()==1) { // limit to one reconstructed particle
            int PID = eventBank.getInt("pid", 0); // PID, kinematic cuts to be improved later?
            if (PID==11) { // check that particle was an electron
                float fitTrack = hitBank.getFloat("trkDoca",0); // 0 = index of particle
                float residuals = hitBank.getFloat("timeResidual",0);
                float docaError = hitBank.getFloat("docaError",0);
                println("Fit track: "+fitTrack+", Residuals: "+residuals+", docaError: "+docaError);
                i++;
            }
        }
    }
}
```

- Check that the relevant banks are present in the event

```java
HipoDataSource reader = new HipoDataSource();
filename = args[0]; // input file at command line
reader.open(filename);

GenericKinematicFitter fitter = new GenericKinematicFitter(11.00);

int i = 0; // counter for desired events (definition coming soon)
int n_events = 10000; //
for(int j=0; j<n_events; j++){ // limit to a certain number of events defined by n_events
// while(reader.hasEvent()==true){ // OR cycle through the entire input file
    HipoDataEvent event = reader.getNextEvent();
    boolean banks_test = true; // check to see if the event has all of the necessary banks present
    if (!(event.hasBank("REC::Particle"))) {
        banks_test = false;
        // println("no event bank"); // warning for no reconstructed event bank
    } else if (!(event.hasBank("TimeBasedTrkg::TBHits"))) {
        banks_test = false;
        // println("no time based tracking banks"); // warning for no TimeBasedTrkg bank
    }
    if (banks_test) { // check that required banks are present
        HipoDataBank eventBank = (HipoDataBank) event.getBank("REC::Particle");
        HipoDataBank hitBank = (HipoDataBank) event.getBank("TimeBasedTrkg::TBHits");
        if (eventBank.rows()==1) { // limit to one reconstructed particle
            int PID = eventBank.getInt("pid", 0); // PID, kinematic cuts to be improved later?
            if (PID==11) { // check that particle was an electron
                float fitTrack = hitBank.getFloat("trkDoca",0); // 0 = index of particle
                float residuals = hitBank.getFloat("timeResidual",0);
                float docaError = hitBank.getFloat("docaError",0);
                println("Fit track: "+fitTrack+", Residuals: "+residuals+", docaError: "+docaError);
                i++;
            }
        }
    }
}
```

- Load banks

```java
HipoDataSource reader = new HipoDataSource();
filename = args[0]; // input file at command line
reader.open(filename);

GenericKinematicFitter fitter = new GenericKinematicFitter(11.00);

int i = 0; // counter for desired events (definition coming soon)
int n_events = 10000; //
for(int j=0; j<n_events; j++){ // limit to a certain number of events defined by n_events
// while(reader.hasEvent()==true){ // OR cycle through the entire input file
    HipoDataEvent event = reader.getNextEvent();
    boolean banks_test = true; // check to see if the event has all of the necessary banks present
    if (!(event.hasBank("REC::Particle"))) {
        banks_test = false;
        // println("no event bank"); // warning for no reconstructed event bank
    } else if (!(event.hasBank("TimeBasedTrkg::TBHits"))) {
        banks_test = false;
        // println("no time based tracking banks"); // warning for no TimeBasedTrkg bank
    }
    if (banks_test) { // check that required banks are present
        HipoDataBank eventBank = (HipoDataBank) event.getBank("REC::Particle");
        HipoDataBank hitBank = (HipoDataBank) event.getBank("TimeBasedTrkg::TBHits");
        if (eventBank.rows()==1) { // limit to one reconstructed particle
            int PID = eventBank.getInt("pid", 0); // PID, kinematic cuts to be improved later?
            if (PID==11) { // check that particle was an electron
                float fitTrack = hitBank.getFloat("trkDoca",0); // 0 = index of particle
                float residuals = hitBank.getFloat("timeResidual",0);
                float docaError = hitBank.getFloat("docaError",0);
                println("Fit track: "+fitTrack+", Residuals: "+residuals+", docaError: "+docaError);
                i++;
            }
        }
    }
}
```

- Require a single reconstructed electron and nothing else (?)

```java
HipoDataSource reader = new HipoDataSource();
filename = args[0]; // input file at command line
reader.open(filename);

GenericKinematicFitter fitter = new GenericKinematicFitter(11.00);

int i = 0; // counter for desired events (definition coming soon)
int n_events = 10000; //
for(int j=0; j<n_events; j++){ // limit to a certain number of events defined by n_events
// while(reader.hasEvent()==true){ // OR cycle through the entire input file
    HipoDataEvent event = reader.getNextEvent();
    boolean banks_test = true; // check to see if the event has all of the necessary banks present
    if (!(event.hasBank("REC::Particle"))) {
        banks_test = false;
        // println("no event bank"); // warning for no reconstructed event bank
    } else if (!(event.hasBank("TimeBasedTrkg::TBHits"))) {
        banks_test = false;
        // println("no time based tracking banks"); // warning for no TimeBasedTrkg bank
    }
    if (banks_test) { // check that required banks are present
        HipoDataBank eventBank = (HipoDataBank) event.getBank("REC::Particle");
        HipoDataBank hitBank = (HipoDataBank) event.getBank("TimeBasedTrkg::TBHits");
        if (eventBank.rows()==1) { // limit to one reconstructed particle
            int PID = eventBank.getInt("pid", 0); // PID, kinematic cuts to be improved later?
            if (PID==11) { // check that particle was an electron
                float fitTrack = hitBank.getFloat("trkDoca",0); // 0 = index of particle
                float residuals = hitBank.getFloat("timeResidual",0);
                float docaError = hitBank.getFloat("docaError",0);
                println("Fit track: "+fitTrack+", Residuals: "+residuals+", docaError: "+docaError);
                i++;
            }
        }
    }
}
```

- Grab residuals, doca, etc. from TimeBasedTrkg bank