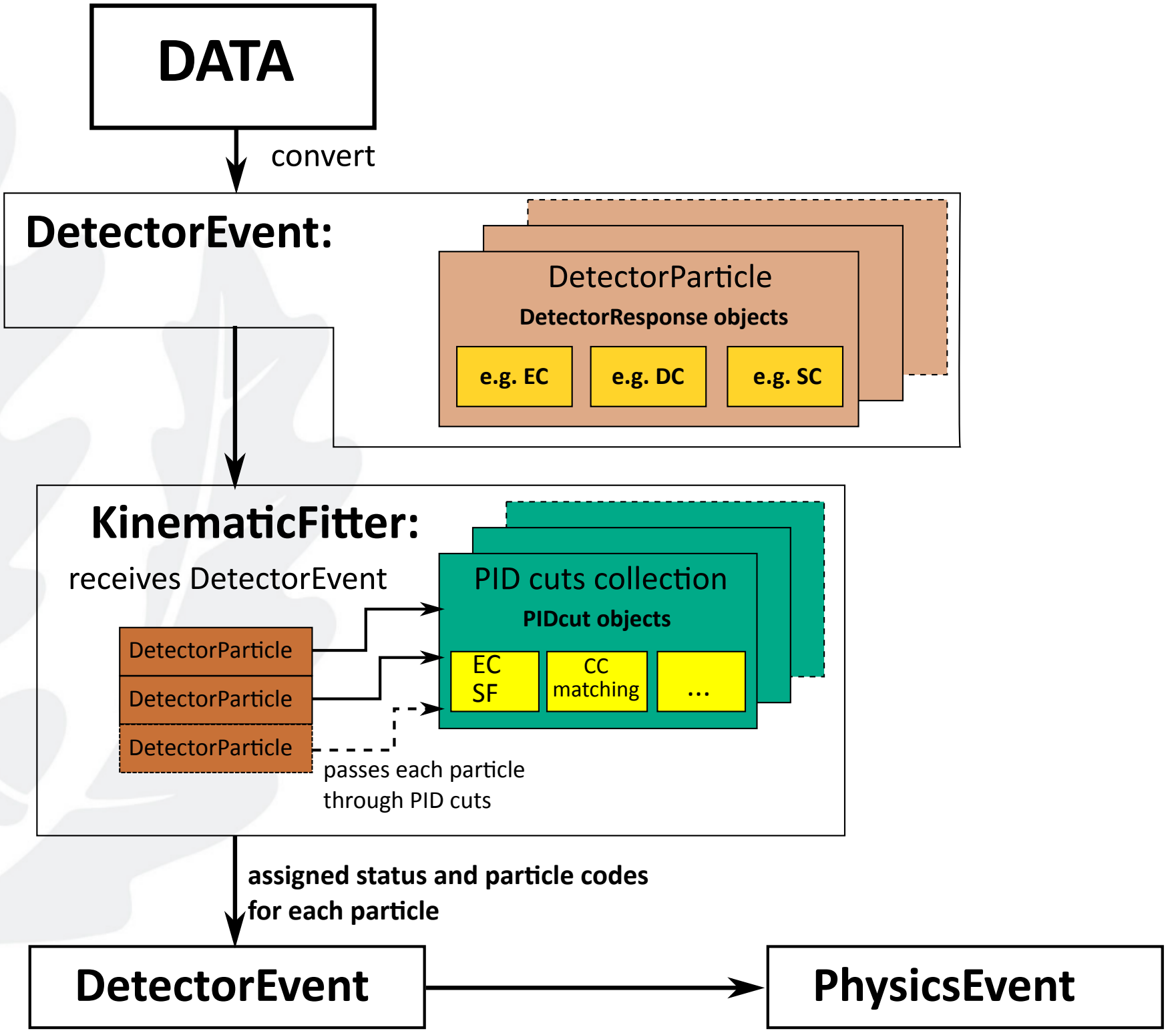
CLAS12 SOFTWARE APPLICATION to eg1-dvcs data

A. Kim

CLAS Collaboration
October 2015


DATA
convert
DetectorEvent:
DetectorParticle
DetectorResponse objects
e.g. EC
e.g. DC
e.g. SC
KinematicFitter:
receives DetectorEvent
PID cuts collection
PIDcut objects
EC
SF
CC
matching
...
DetectorParticle
DetectorParticle
DetectorParticle
passes each particle
through PID cuts
assigned status and particle codes
for each particle
DetectorEvent
PhysicsEvent

Usage:

1. **Download jar file for eg1dvcs experiment:**
   https://github.com/drewkenjo/clas6/blob/master/eg1dvcs/dist/eg1dvcs.jar?raw=true

2. **Move eg1dvcs.jar file to the directory where java can find this jar file:**
   e.g. coatjava/lib/clas

3. **Preliminary eg1dvcs package is ready for use!**

OR

4. **Access the source code on github:**

   // clone my clas6 repo

   >> git clone git@github.com:drewkenjo/clas6.git

Custom PID example

clas6CustomPid.groovy

```
import org.jlab.clas.physics.*
import org.jlab.clas6.*
import org.jlab.evio.clas12.*
import org.root.histogram.H2D
import org.root.pad.TCanvas

def eg1fitter = new CustomKinematicFitter(6);
eg1fitter.addEventProcessor(new eg1dvcs.kenjo.ElectronProcessor());
eg1fitter.addEventProcessor(new eg1dvcs.kenjo.ProtonProcessor());

def eventFilter = new EventFilter("11:2212:X+:X-:Xn");
def reader = new EvioSource();
reader.open(args[0]);

def heSF = new H2D("electron sampling fraction",100,0,6,100,0,0.5)
def hpbeta = new H2D("proton beta",100,0,3,120,0,1.2)

while(reader.hasEvent()==true){
    EvioDataEvent dataev = reader.getNextEvent();
    PhysicsEvent  physev = eg1fitter.getPhysicsEventProcessed(dataev);

    if(eventFilter.isValid(physev)==true){
        def elepart = physev.getParticleByPid(11,0)
        heSF.fill(elepart.p(), elepart.getDetectorParticle().getProperty("sampling fraction"))

        def propart = physev.getParticleByPid(2212,0)
        println propart.getDetectorParticle().propertyString();
        hpbeta.fill(propart.p(), propart.getDetectorParticle().getProperty("beta after cut"))
    }
}

TCanvas c1 = new TCanvas("c1","JROOT Demo",600,800,1,2);
c1.cd(0)
c1.draw(heSF);
c1.cd(1)
c1.draw(hpbeta)
```

Custom PID

clas6CustomPid.groovy

```
1  import org.jlab.clas.physics.*
2  import org.jlab.clas6.*
3  import org.jlab.evio.clas12.*
4  import org.root.histogram.H2D
5  import org.root.pad.TCanvas
6
7  def eg1fitter = new CustomKinematicFitter(6);
8  eg1fitter.addEventProcessor(new eg1dvcs.kenjo.ElectronProcessor());
9  eg1fitter.addEventProcessor(new eg1dvcs.kenjo.ProtonProcessor());
10
11 def eventFilter = new EventFilter("11:2212:X+:X-:Xn");
12 def reader = new EvioSource();
13 reader.open(args[0]);
14
15 def heSF = new H2D("electron sampling fraction",100,0,6,100,0,0.5)
16 def hpbeta = new H2D("proton beta",100,0,3,120,0,1.2)
17
18 while(reader.hasEvent()==true){
19     EvioDataEvent dataev = reader.getNextEvent();
20     PhysicsEvent  physev = eg1fitter.getPhysicsEventProcessed(dataev);
21
22     if(eventFilter.isValid(physev)==true){
23         def elepart = physev.getParticleByPid(11,0)
24         heSF.fill(elepart.p(), elepart.getDetectorParticle().getProperty("sampling fraction"))
25
26         def propart = physev.getParticleByPid(2212,0)
27         println propart.getDetectorParticle().propertyString();
28         hpbeta.fill(propart.p(), propart.getDetectorParticle().getProperty("beta after cut"))
29     }
30 }
31
32 TCanvas c1 = new TCanvas("c1","JROOT Demo",600,800,1,2);
33 c1.cd(0)
34 c1.draw(heSF);
35 c1.cd(1)
36 c1.draw(hpbeta)
```

CustomKinematicFitter accepts the user-defined event processors

ElectronProcessor, ProtonProcessor processors available from user "**kenjo**" from the "**eg1dvcs**" package

Custom PID: electron

ElectronProcessor.java

```java
package org.jlab.clas6.eg1dvcs.kenjo;

import org.jlab.clas.detector.DetectorType;
import org.jlab.clas.physics.*;

/**
 *
 * @author kenjo
 */
public class ElectronProcessor implements IDetectorEventProcessor {

    private IDetectorParticleProcessor esf = new eSamplingFraction();

    @Override
    public void processDetectorEvent(DetectorEvent event) {
        if(event.getParticles().isEmpty())
            return;

        DetectorParticle detpart = event.getParticles().get(0);
        if (detpart.getStatus() == 111) {
            return;
        }

        DetectorResponse scres = detpart.getResponse(DetectorType.SC, 0);
        if (detpart.getCharge()==-1
                && scres!=null
                && esf.processParticle(detpart, event)) {
            event.setStartTime(scres.getTime()-scres.getPath()/30.);
            detpart.setStatus(111);
            detpart.setPid(11);
        }
    }
}
```

package name

EventProcessor interface

list of ParticleProcessors

sequence of conditions PID cuts

Custom PID: sampling fraction

eSamplingFraction.java

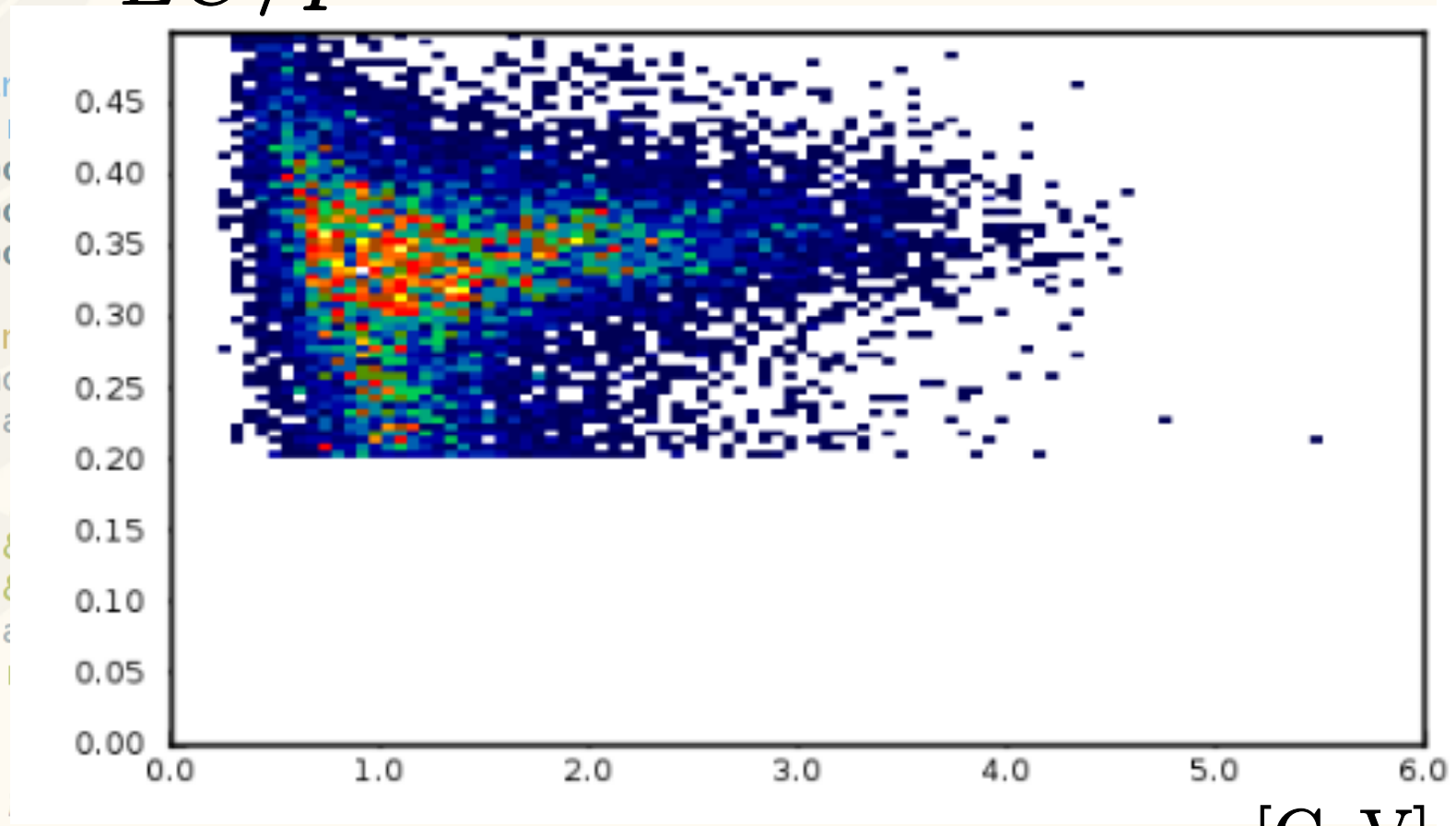
```java
package org.jlab.clas6.eg1dvcs.kenjo;

import org.jlab.clas.detector.DetectorType;
import org.jlab.clas.physics.*;

/**
 *
 * @author kenjo
 */
public class eSamplingFraction implements IDetectorParticleProcessor{
    public boolean processParticle(DetectorParticle detpart, DetectorEvent detevent) {
        DetectorResponse ecin = detpart.getResponse(DetectorType.EC, 0);
        DetectorResponse ecout = detpart.getResponse(DetectorType.EC, 1);
        DetectorResponse ectot = detpart.getResponse(DetectorType.EC, 2);

        if (ecin != null && ecout != null && ectot != null) {
            double momentum = detpart.vector().mag();
            double samplingFraction = ectot.getEnergy() / (momentum-0.12);

            if (ecin.getEnergy() > 0.05
                    && ecout.getEnergy() > 0.05
                    && samplingFraction > 0.2) {
                detpart.addProperty("sampling fraction", samplingFraction);
                return true;
            }
        }
        return false;
    }
}
```

Custom PID: sampling fraction

eSamplingFraction.java

```
1  package org.jlab.clas6.eg1dvcs.kenjo;
2
3  import org.jlab.clas.detector.DetectorType;
4  import org.jlab.clas.physics.*;
5
6  /**
7   *
8   * @author kenjo
9   */
10 public class eSamplin
11     public boolean p
12         DetectorRespo
13         DetectorRespo
14         DetectorRespo
15
16         if (ecin != n
17             double mo
18             double sa
19
20             if (ecin
21                 &
22                 &
23                 detpa
24                 retu
25             }
26         }
27         return false;
28     }
29 }
```

E_{EC}/p



Custom PID: proton

ProtonProcessor.java

```
1  package org.jlab.clas6.eg1dvcs.kenjo;
2
3  import org.jlab.clas.physics.*;
4
5  /**
6   *
7   * @author kenjo
8   */
9  public class ProtonProcessor implements IDetectorEventProcessor {
10
11     public IDetectorParticleProcessor pbetaCut = new pTOFbeta();
12     public IDetectorParticleProcessor eloss = new pEnergyLossCorrection();
13
14     @Override
15     public void processDetectorEvent(DetectorEvent event) {
16         for (DetectorParticle detpart : event.getParticles()) {
17             if (detpart.getStatus() != 111
18                     && detpart.getCharge() == 1
19                     && pbetaCut.processParticle(detpart, event)) {
20
21                 eloss.processParticle(detpart, event);
22                 detpart.setStatus(111);
23                 detpart.setPid(2212);
24             }
25         }
26     }
27 }
```

1

2

3

Custom PID: TOF beta

pTOFbeta.java

```java
package org.jlab.clas6.eg1dvcs.kenjo;

import static java.lang.Math.abs;
import static java.lang.Math.sqrt;
import org.jlab.clas.detector.DetectorType;
import org.jlab.clas.physics.*;

/**
 *
 * @author kenjo
 */

public class pTOFbeta implements IDetectorParticleProcessor{
    @Override
    public boolean processParticle(DetectorParticle detpart, DetectorEvent detevent) {

        DetectorResponse scres = detpart.getResponse(DetectorType.SC, 0);

        if (scres != null) {
            double rsc = scres.getPath();
            double tsc = scres.getTime();
            double beta = rsc/(tsc-detevent.getStartTime())/30;
            detpart.addProperty("beta before cut", beta);

            double mom=detpart.vector().mag();
            double dbeta = beta - mom/sqrt(mom*mom+0.985*0.985);

            if(abs(dbeta)<0.05){
                detpart.addProperty("beta after cut", beta);
                return true;
            }
        }
        return false;
    }
}
```

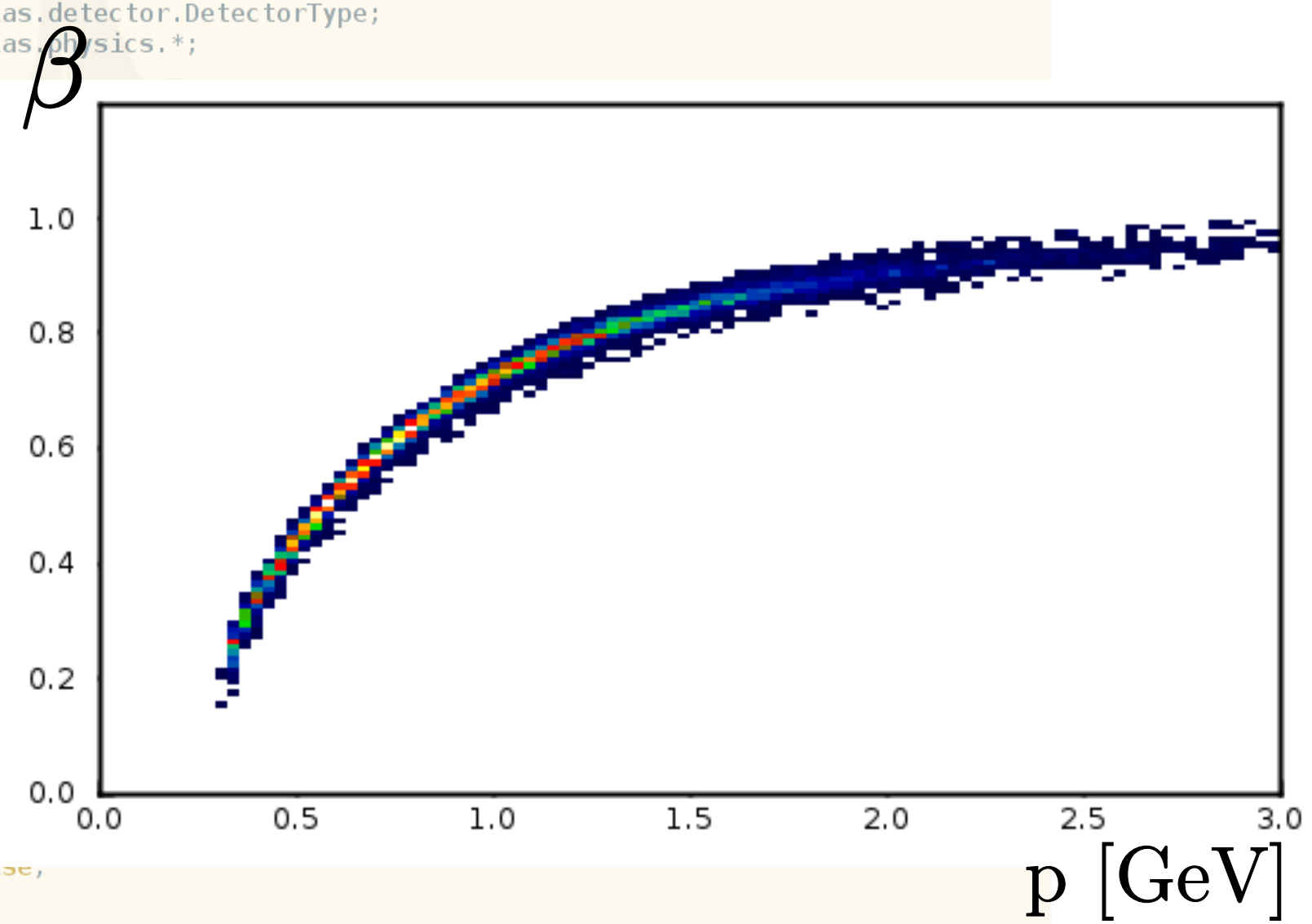
1

2

Custom PID: TOF beta

pTOFbeta.java

```
1  package org.jlab.clas6.eg1dvcs.kenjo;
2
3  import static java.lang.Math.abs;
4  import static java.lang.Math.sqrt;
5  import org.jlab.clas.detector.DetectorType;
6  import org.jlab.clas.physics.*;
7
8  /**
9   *
10  * @author kenjo
11  */
12
13 public class pTO
14     @Override
15     public boole
16
17         Detector
18
19         if (scre
20             doub
21             doub
22             doub
23             detp
24
25             doub
26             doub
27
28             if(a
29
30
31             }
32         }
33         return false;
34     }
35 }
```



Custom PID: energy loss correction

pEnergyLossCorrection.java

```java
package org.jlab.clas6.eg1dvcs.kenjo;

import static java.lang.Math.pow;
import org.jlab.clas.physics.*;

/**
 *
 * @author kenjo
 */

public class pEnergyLossCorrection implements IDetectorParticleProcessor{
    @Override
    public boolean processParticle(DetectorParticle detpart, DetectorEvent detevent) {
        double momentum=detpart.vector().mag();
        momentum *= 0.993 + 0.015/momentum + 0.008/pow(momentum,2)+0.003/pow(momentum,3);
        detpart.vector().setMagThetaPhi(momentum, detpart.vector().theta(), detpart.vector().phi());

        return false;
    }
}
```

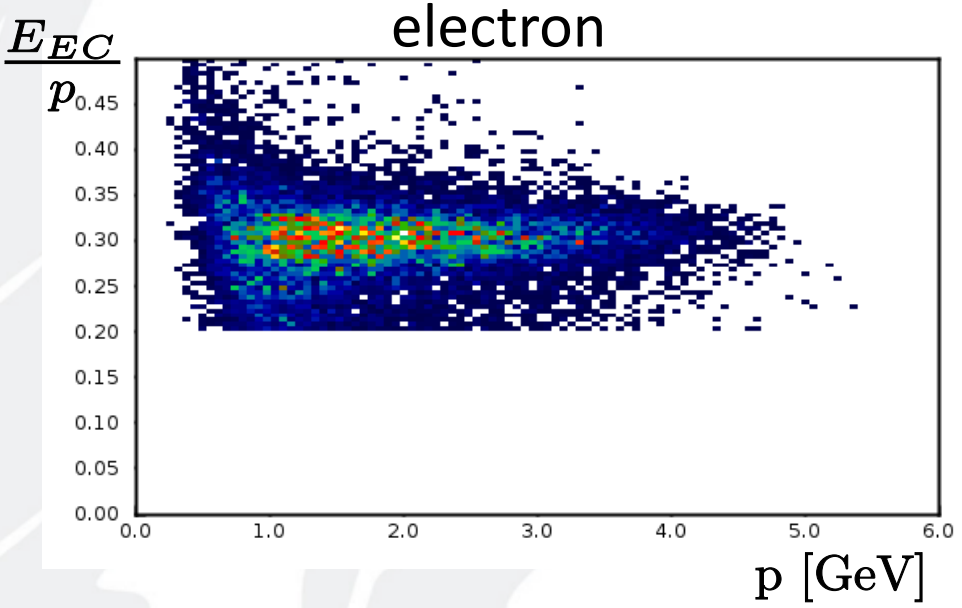
CLAS12 Software approach

- **Short and concise code:**
  readability and easy bug hunting
- **Easy code sharing through "git" system**
- **Unified software approach for CLAS6 and CLAS12 data:**
  access to the detectors and their responses
  descriptive response names (getTime etc...)
- **Short learning curve**

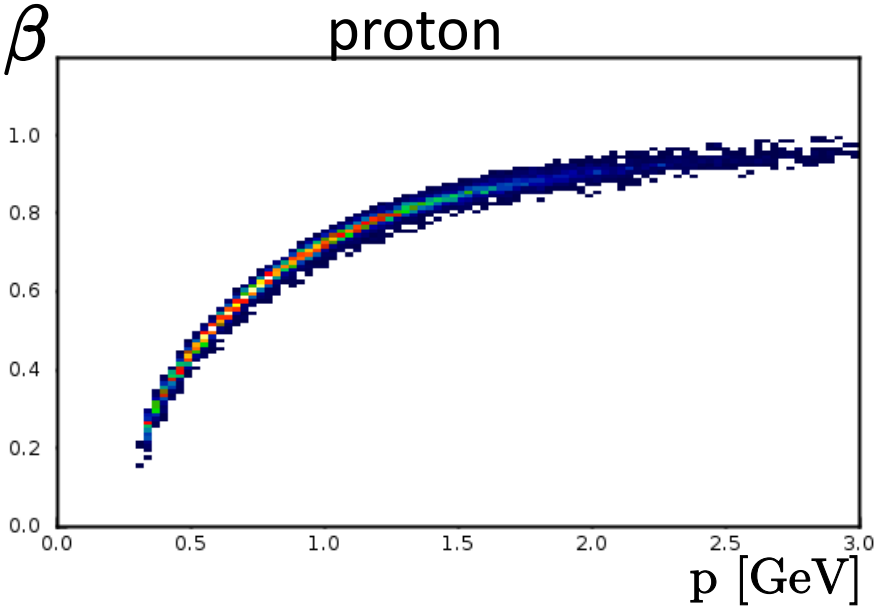
Elastic events

```
def eventFilter = new EventFilter("11:2212:X+:X-:Xn");
```

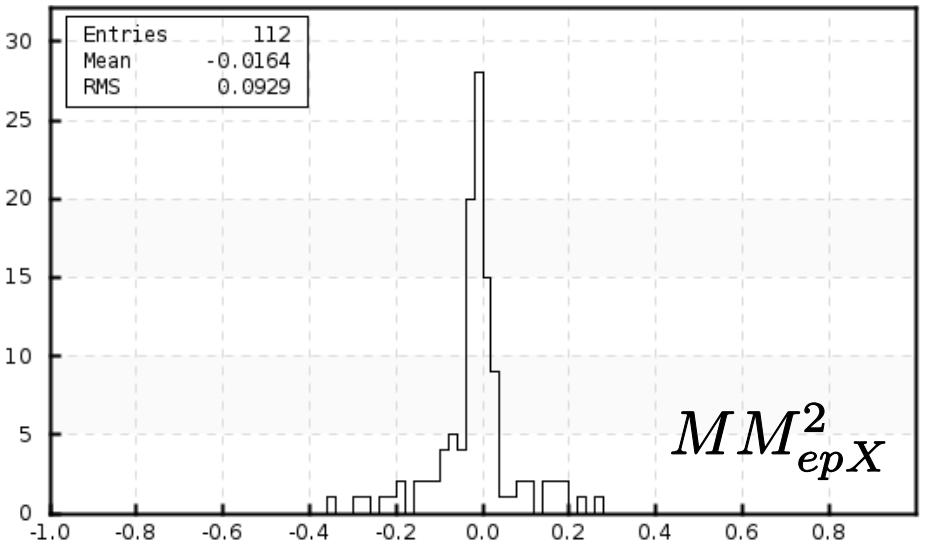
electron



proton




```
def pmiss = physev.getParticle("[b]+[t]-[11]-[2212]");

if(pmiss.get("px")<0.2)
if(pmiss.get("py")<0.2)
if(pmiss.get("pz")<0.2){
```



DONE:

- The CLAS12 software framework is under developement to unify CLAS6 and CLAS12 data analysis
- The sample user code was developed for CLAS6 data and tested on a few runs of eg1dvcs data

TODO:

- Apply CLAS12 software to all runs of eg1dvcs data
- Analyze DVCS, DVπ^0P reactions using CLAS12 software
- Benchmarking of CLAS12 software analysis code
- WRITE DOCUMENTATION!