

org.lcsim FASTMC Modifications

Tim Barklow

SLAC

June 13, 2006

util/DriverAdapter.java
util/lcio/AbstractBlockHandler.java
util/lcio/LCIOWriter.java
util/loop/LCSimLoop.java
mc/fast/MCFast.java
mc/fast/reconstructedparticle/IDResolutionTables.java
mc/fast/reconstructedparticle/MCFastReconstructedParticle.java
mc/fast/reconstructedparticle/MCFastReconstructedParticleDriver.java
mc/fast/tracking/DocaTrackParameters.java
mc/fast/tracking/LookupTable.java
mc/fast/tracking/MCFastTracking.java
mc/fast/tracking/SimpleTables.java
mc/fast/tracking/SmearTrackSimple.java
mc/fast/cluster/ronan/ClusterResolutionTables.java
mc/fast/cluster/ronan/MCFastRonan.java
mc/fast/cluster/ronan/ReconCluster.java

util/lcio/AbstractBlockHandler.java

11d10

```
< import org.lcsim.event.ReconstructedParticle;
```

50d48

```
<      //      System.out.println(" AbstractBlockHandler out= "+out+" flags=
"+flags+" collection.size()= "+collection.size());
```

53d50

```
<      //      if(md.getName().equals("MCFastReconstructedParticles"))
```

```
System.out.println(" AbstractBlockHandler
```

```
( (ReconstructedParticle)element).getCharge()=
```

```
"+"+(ReconstructedParticle)element).getCharge()+" getEnergy()=
```

```
"+"+(ReconstructedParticle)element).getEnergy());
```

59c56

```
< }
```

```
> }
```

```
\ No newline at end of file
```

```

4,5d3
< import static java.lang.Math.sqrt;
< import static java.lang.Math.pow;
9,15d6
<     private boolean JETParameterization;
<     private double JETResolution;
<     private double JETHadDegradeFraction;
<     private double JETEMEnergyFraction;
<     private double JETHadEnergyFraction;
<     private double Lambda_j;
<
37,42d27
<     JETParameterization =
Boolean.parseBoolean(set.getString("JETParameterization"));
<     JETResolution = set.getDouble("JETResolution");
<     JETHadDegradeFraction = set.getDouble("JETHadDegradeFraction");
<     JETEMEnergyFraction = set.getDouble("JETEMEnergyFraction");
<     JETHadEnergyFraction = set.getDouble("JETHadEnergyFraction");
<
62,71d46
<     if (JETParameterization)
<     {
<         EMConstantTerm=0.;
<         HADConstantTerm=0.;
<         Lambda_j=(pow(JETResolution,2)-
JETEMEnergyFraction*pow(EMResolution,2)-
JETHadEnergyFraction*pow(HADResolution,2))
<             /((1.-
JETHadDegradeFraction)*JETEMEnergyFraction*pow(EMResolution,2)+JETHadDegradeFrac
tion*JETHadEnergyFraction*pow(HADResolution,2));
<         EMResolution*=sqrt(1.+Lambda_j*(1.-JETHadDegradeFraction));
<         HADResolution*=sqrt(1.+Lambda_j*JETHadDegradeFraction);
<         System.out.println(" JETParameterization settings   Lamda_j=
"+Lambda_j+" EMResolution= "+EMResolution+" HADResolution= "+HADResolution);

```

mc/fast/tracking/DocaTrackParameters.java

18c18

< * @version \$Id: DocaTrackParameters.java,v 1.1 2006/05/26 07:21:55 timb Exp \$

> * @version \$Id: DocaTrackParameters.java,v 1.5 2005/08/20 23:24:14 tonyj Exp

\$

299c299

< if ((refPoint.x() != 0.) || (refPoint.y() != 0.))

> if ((refPoint.x() != 0.) && (refPoint.y() != 0.))

333c333

< if ((refPoint.x() != 0.) || (refPoint.y() != 0.))

> if ((refPoint.x() != 0.) && (refPoint.y() != 0.))

358c358

< if ((refPoint.x() != 0.) || (refPoint.y() != 0))

> if ((refPoint.x() != 0.) && (refPoint.y() != 0))

util/DriverAdapter.java

16,19c16

```
<     private Driver driver;  
<     private long nskip;  
<     private long neventheader;  
<
```

<

```
>     private Driver driver;
```

22,26d18

```
<         this(driver, 0);  
<     }
```

<

```
<     public DriverAdapter(Driver driver, long nskip)
```

```
<     {
```

28,29d19

```
<         this.nskip = nskip ;  
<         neventheader = 0;
```

50,53c40

```
<         if (event instanceof EventHeader) {  
<             neventheader++;  
<             if(neventheader > nskip) driver.process((EventHeader) event);  
<         }
```

```
>         if (event instanceof EventHeader) driver.process((EventHeader) event);
```

mc/fast/reconstructedparticle/IDResolutionTables.java

21,26c21,23

```
<     private double ElectronEff;  
<     private double MuonEff;  
<     private double ProtonEff;  
<     private double KaonEff;  
<     private double NeutronEff;  
<     private double WtChgTrkCal;
```

```
>     private double  ElectronEff;  
>     private double  MuonEff;  
>     private double  NeutronEff;
```

33,34d29

```
<         ProtonEff = set.getDouble("Proton");  
<         KaonEff = set.getDouble("Kaon");
```

36d30

```
<         WtChgTrkCal = set.getDouble("wt_charged_track_calorimeter_energy");
```

46,56c40

```
<         return MuonEff;  
<     }
```

<

```
<     public double getProtonEff()  
<     {
```

<

```
<         return ProtonEff;
```

<

<

```
<     public double getKaonEff()  
<     {
```

<

```
<         return KaonEff;
```

```
>         return ElectronEff;
```

61c45

```
<         return NeutronEff;
```

```
>         return ElectronEff;
```

64,67d47

```
<     public double getWtChgTrkCal()  
<     {
```

<

```
<         return WtChgTrkCal;
```

<

```
<     }
```

util/lcio/LCIOWriter.java

```
8d7
< import java.util.ArrayList;
28d26
< private List<String> listIgnore = new ArrayList<String>();
38,47d35
< public LCIOWriter(File file, List<String> listIgnore) throws IOException
< {
<     this(file);
<     this.listIgnore = listIgnore;
< }
< public LCIOWriter(String file, List<String> listIgnore) throws IOException
< {
<     this(file);
<     this.listIgnore = listIgnore;
< }
69d56
< // System.out.println(" LCIOWriter headerOnly md.getName()=
"+md.getName()+" type.getName()= "+type.getName());
72c59
<     else if(!listIgnore.contains(md.getName()))
blocks.put(md.getName(),bh.getType());
---
>     else blocks.put(md.getName(),bh.getType());
78,82c65,66
< // System.out.println(" LCIOWriter entry.getKey()=
"+entry.getKey()+" entry.getValue()= "+entry.getValue());
<     if(!listIgnore.contains(entry.getKey())) {
<         out.writeString(entry.getKey());
<         out.writeString(entry.getValue());
<     }
---
>         out.writeString(entry.getKey());
>         out.writeString(entry.getValue());
96,101c80,81
< // System.out.println(" LCIOWriter md.getName()=
"+md.getName()+" type.getName()= "+type.getName());
<     if(!listIgnore.contains(md.getName())) {
<         LCIOBlockHandler bh = manager.handlerForClass(type);
<         // System.out.println(" LCIOWriter bh= "+bh);
<         if (bh != null) bh.writeBlock(writer,collection,md);
<     }
---
>         LCIOBlockHandler bh = manager.handlerForClass(type);
>         if (bh != null) bh.writeBlock(writer,collection,md);
```


util/loop/LCSimLoop.java

23,27d22

< this(0);

< }

<

< public LCSimLoop(long nskip)

< {

30c25

< super.addRecordListener(new DriverAdapter(top, nskip));

> super.addRecordListener(new DriverAdapter(top));

mc/fast/tracking/LookupTable.java

```
67d66
<      //      System.out.println(" index1= "+index1+" index2= "+index2);
92,95d90
<      //      System.out.print("Looking for "+value+" in [");
<      //      for (int i=0; i<key.length; i++) System.out.print(key[i]+",");
<      //      System.out.print("] ");
<      //      System.out.println(" key.length= "+key.length);
98d92
<      //      System.out.println("Interpolation out of range: lower: "+value+"
< "+key[0]);
102,104d95
<      //      else if(value >= key[key.length-1]){
<      //      System.out.println("Interpolation out of range: upper: "+value+"
>= "+key[key.length-1]);
<      //      }
109,110c100
<      // System.out.println("pos= "+pos);
<      return Math.min(pos, key.length-2);
---
>      return pos;
114,115c104
<      //      System.out.println("-pos-2= "+(-pos-2));
<      return Math.min(-pos-2, key.length-2);
---
>      return -pos - 2;
124d112
<      //      System.out.println("Interpolation out of range: upper: "+value+" >=
"+key[key.length-1]);
```

mc/fast/MCFast.java

22,27d21

```
<    public MCFast(boolean beamSpotConstraint, boolean simple, long seed)
<    {
<        this(beamSpotConstraint, simple);
<        getRandom().setSeed(seed);
<    }
<
```

mc/fast/reconstructedparticle/MCFastReconstructedParticle.java

```
7d6
< import hep.physics.vec.VecOp;
16d14
< import org.lcsim.mc.fast.tracking.ReconTrack;
20d17
< import static java.lang.Math.abs;
33,34d29
<     private double e_track;
<     private double e_reco;
36d30
<     private Hep3Vector p3_track;
43,45c37
<     private BasicHepLorentzVector p_reco = new BasicHepLorentzVector();
<     private BasicHepLorentzVector p_track = new BasicHepLorentzVector();
<     private static long iprint=0;
---
>     private BasicHepLorentzVector _fourVec = new BasicHepLorentzVector();
47c39
<     public MCFastReconstructedParticle(Track t, ParticleType type, Particle p,
Cluster assoc_c, double wcal)
---
>     public MCFastReconstructedParticle(Track t, ParticleType type, Particle p)
49,50d40
<     iprint++;
<     if(iprint < 200) System.out.println(" PDGID= "+type.getPDGID()+
t.getPX, ...= "+t.getPX()+" "+t.getPY()+" "+t.getPZ());
52a43,44
>     double e =
sqrt(t.getPX()*t.getPX()+t.getPY()*t.getPY()+t.getPZ()*t.getPZ() + _mass*_mass);
>     _fourVec.setV3(e, t.getPX(), t.getPY(), t.getPZ());
54,82c46,49
<     // Use true point of origin for reference point for now.
<     _referencePoint = p.getOrigin();
<     e_track =
sqrt(((ReconTrack)t).getDocaMomentumVec(_referencePoint).magnitudeSquared()+
_mass*_mass);
<     p_track.setV3(e_track,
((ReconTrack)t).getDocaMomentumVec(_referencePoint));
<     p3_track = p_track.v3();
<     if (iprint < 200)
<     {
<         if (assoc_c != null)
<         {
<             System.out.println(" PDGID= "+type.getPDGID()+" e_track=
"+e_track+" e_assoc_clus= "+assoc_c.getEnergy());
<             System.out.println(" PDGID= "+type.getPDGID()+"
_referencePoint= "+_referencePoint.x()+" "+_referencePoint.y()+"
+_referencePoint.z());
<             System.out.println(" PDGID= "+type.getPDGID()+" p3_track=
"+p3_track.x()+" "+p3_track.y()+" "+p3_track.z());
<         }
<     }
<     else
<     {
<         System.out.println(" assoc_c = null PDGID= "+type.getPDGID());
<     }
<     }
```

```

mc/fast/reconstructedparticle/MCFastReconstructedParticleDriver.java

8,9d7
< import java.util.Map;
< import java.util.HashMap;
45,48d42
<     private ParticleType pplus;
<     private ParticleType pminus;
<     private ParticleType kplus;
<     private ParticleType kminus;
70,73d63
<     pplus = ppp.get(2212);
<     pminus = ppp.get(-2212);
<     kplus = ppp.get(321);
<     kminus = ppp.get(-321);
83,85c73,75
<     ConditionsSet idconditions =
getConditionsManager().getConditions("IDEfficiency");
<     idconditions.addConditionsListener(this);
<     IDEff = new IDResolutionTables(idconditions);
---
>     ConditionsSet conditions =
getConditionsManager().getConditions("IDEfficiency");
>     conditions.addConditionsListener(this);
>     IDEff = new IDResolutionTables(conditions);
90,103d79
<     List<Track> tracks = event.getTracks();
<     List<Cluster> clusters = event.getClusters();
<
<     // Set up Track-Cluster association; for now cheat using MCParticle
<     Map<Particle, Track> m_pt = new HashMap<Particle, Track>();
<     Map<Particle, Cluster> m_pc = new HashMap<Particle, Cluster>();
<     Map<Cluster, Track> m_ct = new HashMap<Cluster, Track>();
<     Map<Track, Cluster> m_tc = new HashMap<Track, Cluster>();
<
<     for(Track t : tracks) m_pt.put(((ReconTrack)t).getMCParticle(),t);
<     for(Cluster c : clusters) m_pc.put((c instanceof ReconEMCluster ?
((ReconEMCluster)c).getMCParticle() : ((ReconHADCluster)c).getMCParticle() ),c);
<     for(Track t : tracks)
m_tc.put(t,m_pc.get(((ReconTrack)t).getMCParticle()));
<     for(Cluster c : clusters) m_ct.put(c,m_pt.get(c instanceof ReconEMCluster
? ((ReconEMCluster)c).getMCParticle() : ((ReconHADCluster)c).getMCParticle() ));
<
105a82
>     List<Track> tracks = event.getTracks();
116c93
<         // charged track id
---
>         // electrons and muons are special
125,132d101
<         else if((abs(pdgid)== 2212) && (rand.nextDouble() <
IDEff.getProtonEff()))
<         {
<             type = rt.getCharge() > 0 ? pplus : pminus;
<         }
<         else if((abs(pdgid)== 321) && (rand.nextDouble() <
IDEff.getKaonEff()))

```

```

mc/fast/cluster/ronan/MCFastRonan.java

25d24
< private final static int ElecID = 11;
27d25
< private final static int MuID = 13;
35d32
< private static long iprint=0;
48c45
< List<Cluster> cl = new ArrayList<Cluster>();
---
> List cl = new ArrayList();
84c81
< if (PDGID == PhotonID || PDGID == ElecID)
---
> if (PDGID == PhotonID)
87,92c84,85
< // double thing = (1 - 1 / ( 1 + Math.exp( (E-
clusterParm.getEMOnset()*clusterParm.getEMSharpness() ) ));
< //if (rand.nextDouble() > thing)
< //{
< // continue;
< //}
< if (E < clusterParm.getEMOnset())
---
> double thing = (1 - 1 / ( 1 + Math.exp( (E-
clusterParm.getEMOnset()*clusterParm.getEMSharpness() ) ));
> if (rand.nextDouble() > thing)
106c99
< else if (PDGID != MuID)
---
> else if (charge == 0)
110,115c103,104
< //double thing = (1 - 1 / ( 1 + Math.exp( (E-
clusterParm.getHADOnset()*clusterParm.getHADSharpness() ) ));
< //if (rand.nextDouble() > thing)
< //{
< //continue;
< //}
< if (E < clusterParm.getHADOnset())
---
> double thing = (1 - 1 / ( 1 + Math.exp( (E-
clusterParm.getHADOnset()*clusterParm.getHADSharpness() ) ));
> if (rand.nextDouble() > thing)
117c106
< continue;
---
> //continue;
121c110
< continue;
---
> //continue;
127,138d115
< double neg_energy_total = 0.;
< double pos_energy_weight_total = 0.;
< for (Cluster rcl : cl )
< {

```

mc/fast/tracking/MCFastTracking.java

```
15d14
< import org.lcsim.particle.ParticleUtilities;
23,26d21
<   private List<Particle> bquark_list = new ArrayList<Particle>();
<   private List<Particle> anti_bquark_list = new ArrayList<Particle>();
<   private Particle cquark;
<   private Particle anti_cquark;
32,33d26
<   private static int nprint = 0 ;
<   private static final int nprint_max = -2 ;
88,90c81,83
<       ConditionsSet simpleconditions =
getConditionsManager().getConditions("SimpleTrack");
<       simpleconditions.addConditionsListener(this);
<       SmTbl = new SimpleTables(simpleconditions);
---
>       ConditionsSet conditions =
getConditionsManager().getConditions("SimpleTrack");
>       conditions.addConditionsListener(this);
>       SmTbl = new SimpleTables(conditions);
96,100c89
<       List<Track> trackList = new ArrayList<Track>();
<       bquark_list.clear();
<       anti_bquark_list.clear();
<       cquark=null;
<       anti_cquark=null;
---
>       List trackList = new ArrayList();
104,114d92
<       if(Math.abs(p.getPDGID()) == 5) {
<           boolean addp;
<
<           if(p.getParents().size() == 0) addp = false;
<           else addp=Math.abs(((Particle)p.getParents().get(0)).getPDGID()) !=
5;
<
<           if(p.getPDGID() == 5 && addp) bquark_list.add(p);
<           if(p.getPDGID() == -5 && addp) anti_bquark_list.add(p);
<       }
<       if(p.getPDGID() == 4) cquark=p;
<       if(p.getPDGID() == -4) anti_cquark=p;
145,167d122
<       nprint++;
<       if(nprint <= nprint_max) {
<
<           for(Particle bquark : bquark_list) {
<               System.out.println(" MCFastTracking nprint= "+nprint+"
bquark.getPDGID= "+bquark.getPDGID()+" bquark.getGeneratorStatus=
"+bquark.getGeneratorStatus()
<                   +" ((Particle)bquark.getParents().get(0)).getPDGID=
"+((Particle)bquark.getParents().get(0)).getPDGID());
<               ParticleUtilities.dumpParticleHierarchy(bquark);
<           }
<       for(Particle anti_bquark : anti_bquark_list) {
```

```

mc/fast/cluster/ronan/ReconCluster.java

18,19d17
<   protected double neg_energy;
<   protected double sigma;
24d21
<   private static long iprint=0;
38,58d34
<   public double getNegEnergy()
<   {
<       return neg_energy;
<   }
<
<   public double getSigma()
<   {
<       return sigma;
<   }
<
<   public void adjustEnergy(double neg_energy_total, double
pos_energy_weight_total)
<   {
<       iprint++;
<       if(iprint < -200) System.out.println("
min(sigma,energy)+"Math.min(sigma,energy)+" ratio=
"+(Math.min(sigma,energy)/pos_energy_weight_total)+" before adjust energy=
"+energy);
<
energy+=neg_energy_total*Math.min(sigma,energy)/pos_energy_weight_total;
<
<       if (energy <= mcp.getMass()) energy=mcp.getMass()+Double.MIN_VALUE;
<
<       if(iprint < -200) System.out.println(" neg_energy_total=
"+neg_energy_total+" after adjust energy= "+energy);
<   }
<
74c50
<       sigma = ((a / Math.sqrt(E)) + b) * E;
---
>       double sigma = ((a / Math.sqrt(E)) + b) * E;
75a52,54
>       energy = 0;
>       while (energy <= mcp.getMass())
>       {
77,85c56
<       if (energy <= mcp.getMass())
<       {
<           neg_energy=energy-mcp.getMass();
<           energy=mcp.getMass()+Double.MIN_VALUE;
<       }
<       else
<       {
<           neg_energy=0.;
<       }
---
>   }

```


mc/fast/tracking/SimpleTables.java

13,18c13,14

```
< private double ConstantTerm;
< private double ThetaTerm;
< private double TanLambdaErrorScale;
< private double PhiErrorScale;
< private double D0ErrorScale;
< private double Z0ErrorScale;
```

```
> private double ConstantTerm;
> private double ThetaTerm;
```

25,28d20

```
< TanLambdaErrorScale = set.getDouble("TanLambdaErrorScale");
< PhiErrorScale = set.getDouble("PhiErrorScale");
< D0ErrorScale = set.getDouble("D0ErrorScale");
< Z0ErrorScale = set.getDouble("Z0ErrorScale");
```

40,59d31

```
< public double getTanLambdaErrorScale()
< {
<     return TanLambdaErrorScale;
< }
<
< public double getPhiErrorScale()
< {
<     return PhiErrorScale;
< }
<
< public double getD0ErrorScale()
< {
<     return D0ErrorScale;
< }
<
< public double getZ0ErrorScale()
< {
<     return Z0ErrorScale;
< }
<
```

```

mc/fast/tracking/SmearTrackSimple.java

3,6d2
< import Jama.EigenvalueDecomposition;
< import Jama.Matrix;
<
< import org.lcsim.mc.fast.tracking.SimpleTables;
8c4,5
<
---
> import org.lcsim.mc.fast.tracking.SimpleTables;
> import org.lcsim.util.aida.AIDA;
11c8,9
< * @author T. Barklow
---
> *
> * @author Daniel
13,14c11,14
< class SmearTrackSimple
< {
---
> public class SmearTrackSimple
> {
>
> /** Creates a new instance of SmearTrackSimple */
16,19c16,21
< * Smear track parameters according to modified version of track's stored
error matrix.
< *
< * @see TrackParameters
< */
---
> * Smear track parameters according to the track's stored error matrix.
> *
> * @see TrackParameters
> */
> private static AIDA aida = AIDA.defaultInstance();
>
22,110c24,84
<
< final double errScale=0.0001 ;
< final double eMScale=1.e14 ;
< // get copy of error matrix and prepare for modification
< Matrix eM = Matrix.constructWithCopy(noSmear.getErrorMatrix());
< double[] errscale = { SmTbl.getD0ErrorScale(), SmTbl.getPhiErrorScale(),
1., SmTbl.getZ0ErrorScale(), SmTbl.getTanLambdaErrorScale() };
< double[] oldDiagErr = new double[5];
< double[] newDiagErr = new double[5];
< for(int i=0; i<5; i++) {
< oldDiagErr[i]= Math.sqrt(noSmear.getErrorMatrix()[i][i]);
< if(i == 2) {
< double th = Math.atan(1/(noSmear.getTanL()));
< double a = SmTbl.getConstantTerm();
< double b = SmTbl.getThetaTerm()/(pt*Math.sin(th));
< newDiagErr[i]= pt * Math.sqrt(a*a + b*b);
< }
< else {

```