中国科学技术大学
University of Science and Technology of China

BESIII介绍与Rhopi程序分析

参考马腾师兄的入门分析教程

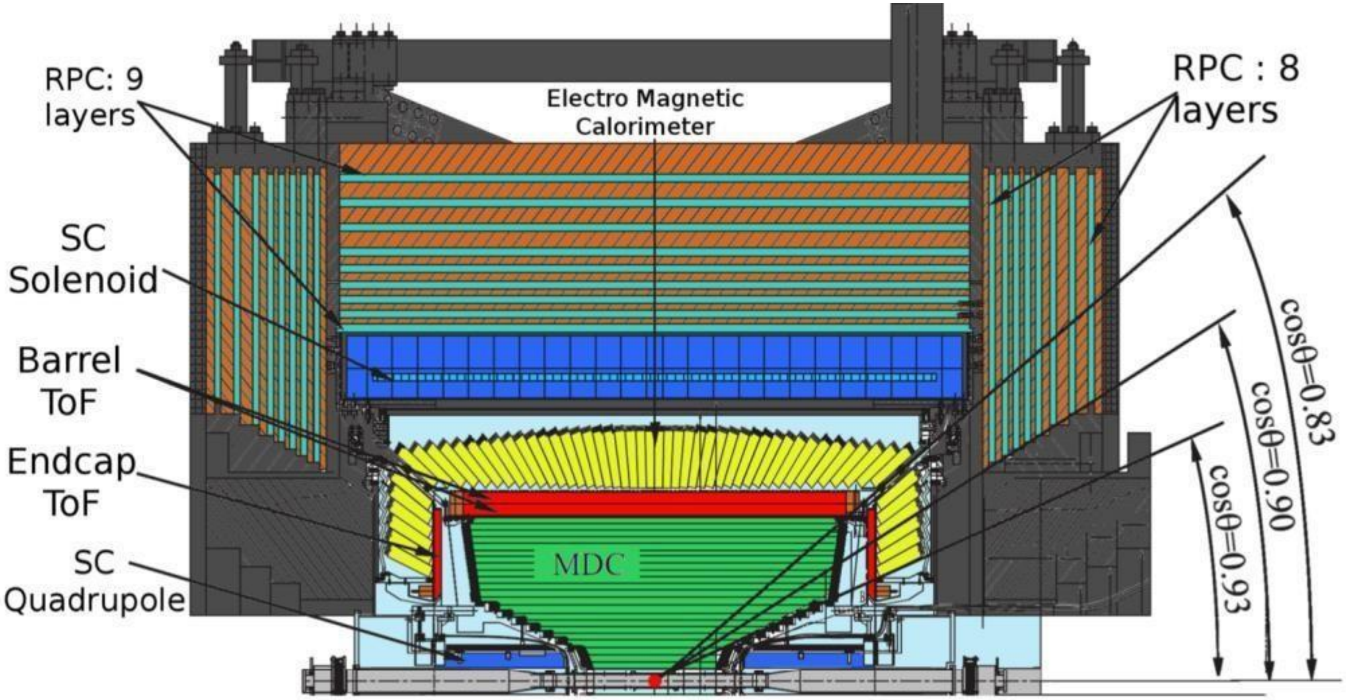
2024年7月30日

一、BESIII 探测器介绍

Beijing Electron Positron Collider II

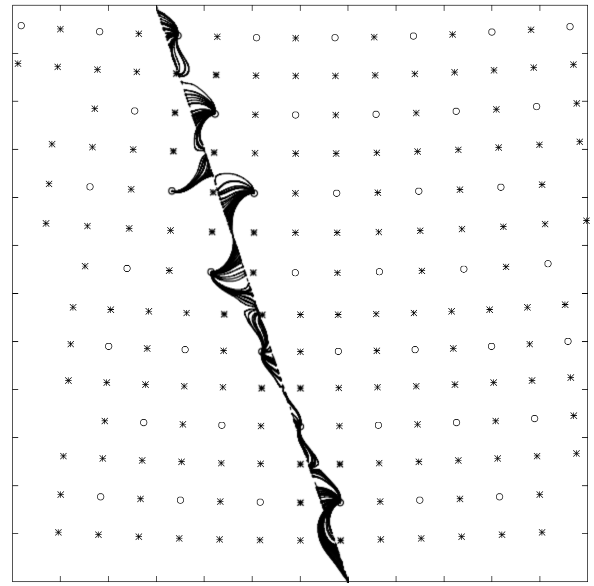
BESIII 探测器简介

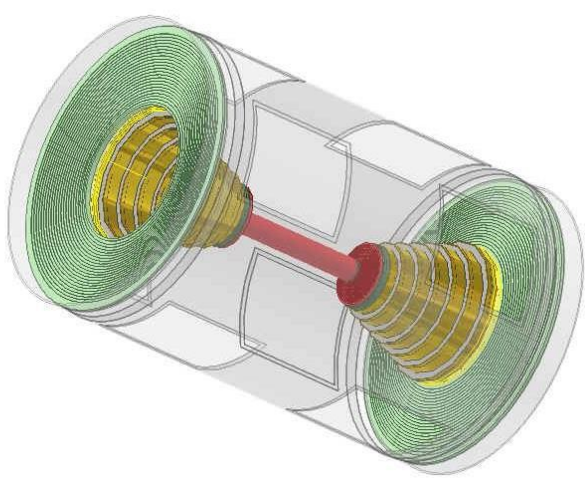
- **Main Draft Chamber (MDC) :** 主漂移室。测量带电粒子的径迹以及电离能损
- **Time of Flight (TOF) :** 飞行时间探测器。测量带电粒子的飞行时间
- **ElectroMagnetic Calorimeter (EMC) :** 电磁量能器。测量电磁型粒子能量。$(e^{\pm}, \gamma)$
- **Super Conducting Solenoid :** 超导磁铁。提供1T磁场。
- **Muon Counter (MUC) :** 测量μ的位置，鉴别μ/π。

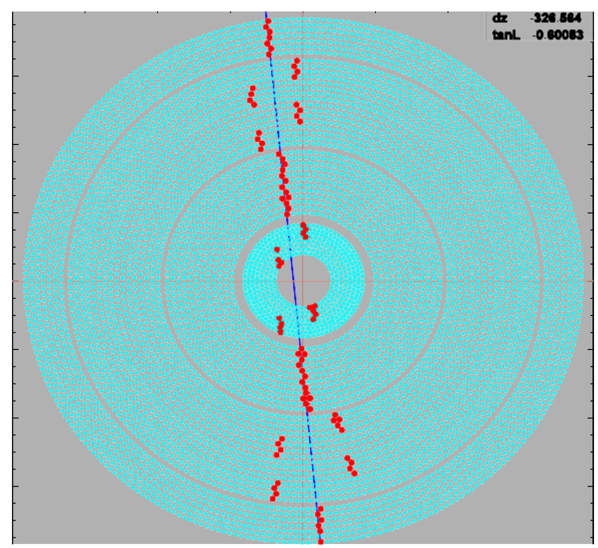


漂移室工作原理

- 带电粒子穿过漂移室，通过电离激发产生电子—离子对，电子在电场下向阳极丝漂移感应信号。
- BESIII的漂移室拉有直丝与斜丝，分别定位径迹的横向与纵向。
- 通过寻迹算法将探测器上的击中转化为径迹信息。

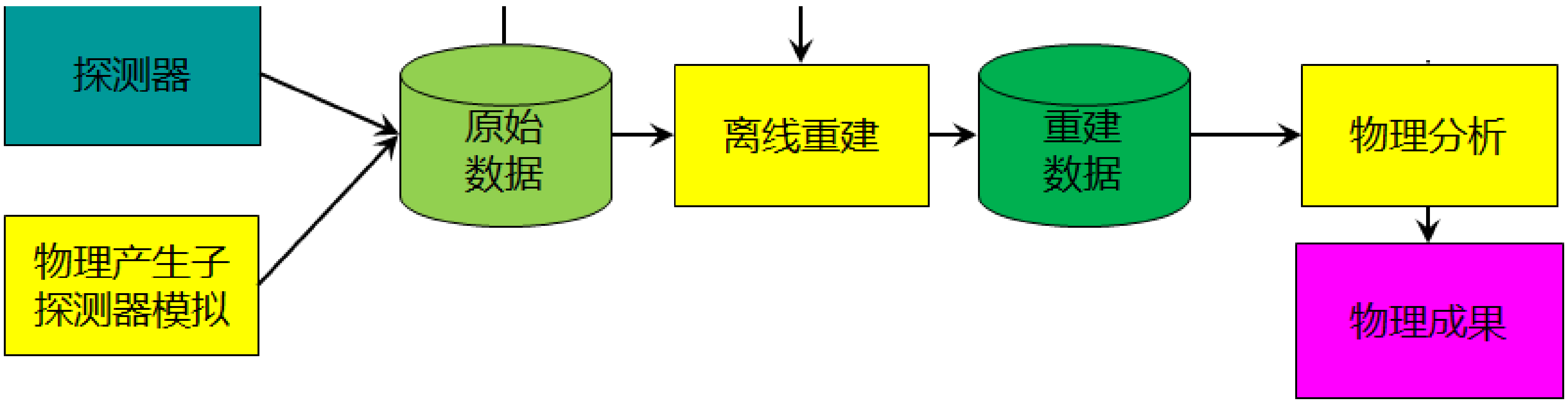




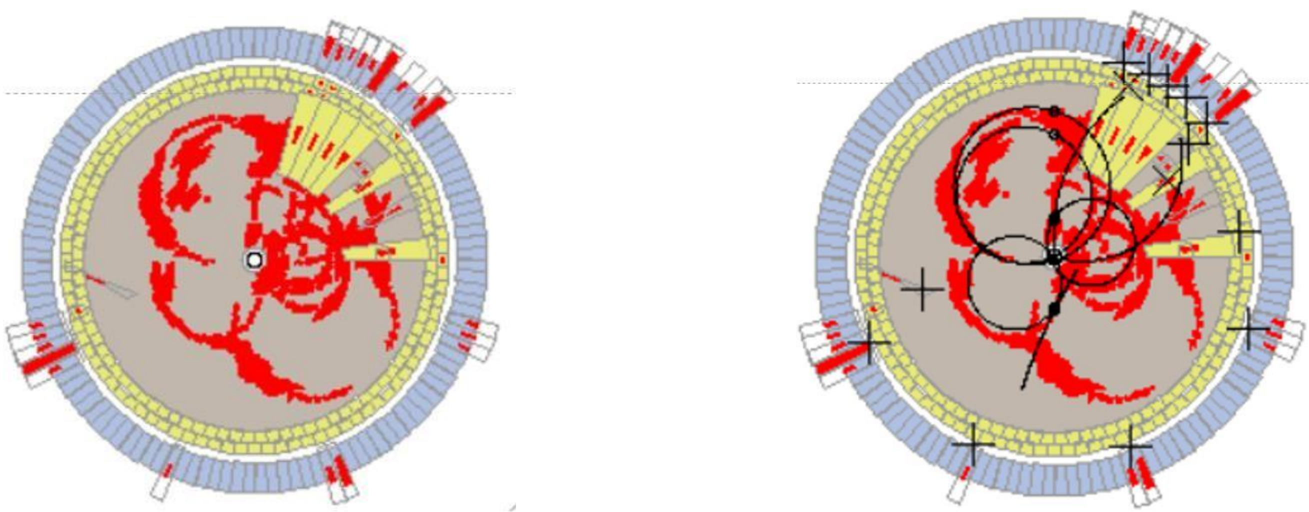


如何测量的呢？

- BESIII实验上可探测的末态粒子：$e, \mu, \pi, K, p, \ \gamma$



- 原始数据：探测器给出的电子学信号（包含时间和幅度等信息）
- 重建数据：通过重建算法，得到粒子的动量、能量等物理量

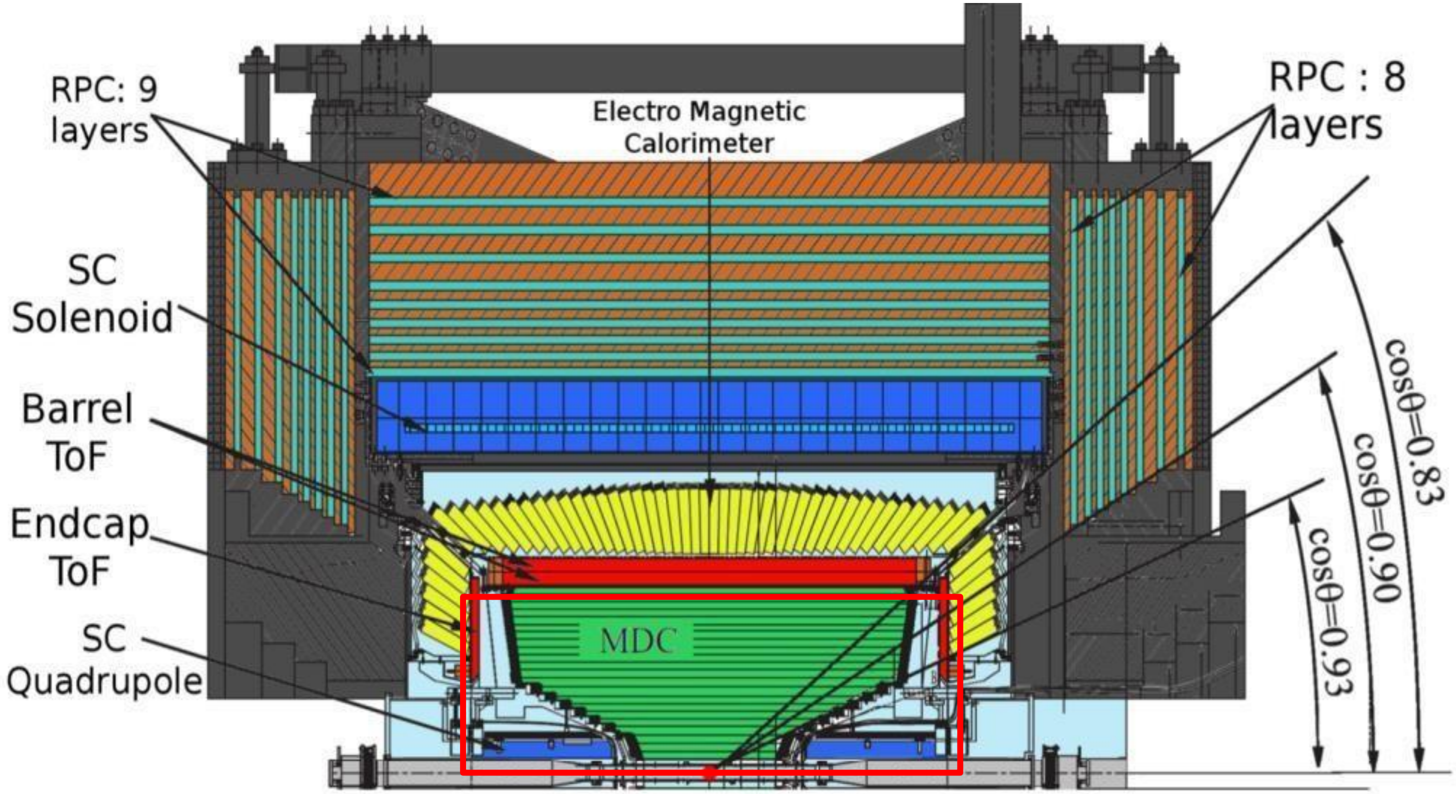


需要测量哪些物理量？

- **计数和计数率**，即粒子通量和流强
- **时间**，粒子到达探测器的时间或相对于某一时刻的时间间隔
- **能量**，粒子(特别是中性粒子)的能量
- **动量**，主要指带电粒子动量
- **位置**，粒子进入探测器的位置，带电粒子飞行径迹，中性粒子的位置与飞行方向
- **粒子的分辨**，通常使用dE/dx与TOF信息联合鉴别

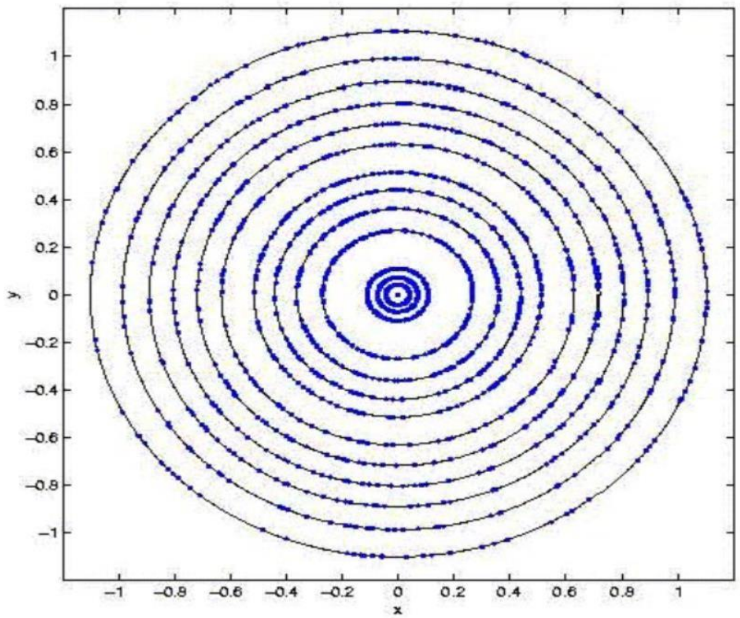
带电粒子的重建

- 带电粒子径迹的描述方法
- MDC工作原理
- 寻迹算法与径迹拟合

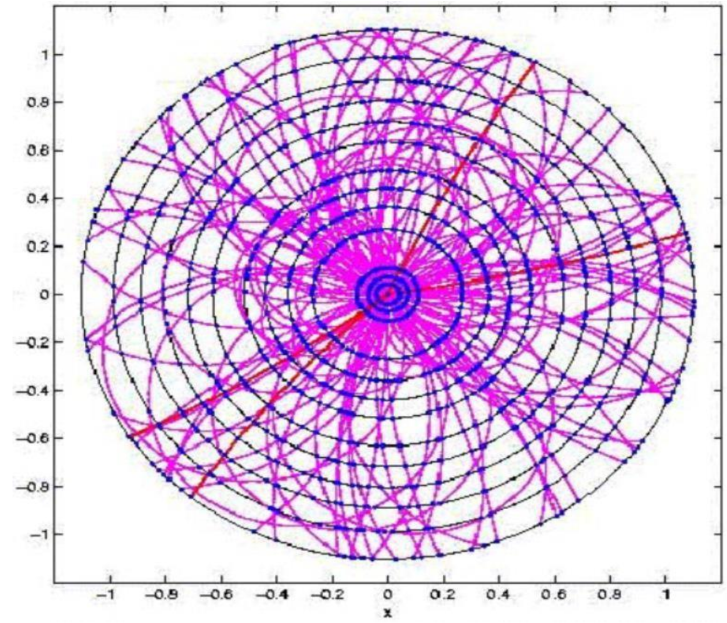


寻迹与拟合

探测器击中信息



重建的径迹



寻迹：　将所有击中分类为一系列击中集合

- 局域寻迹算法
- 全局寻迹算法

拟合：得到径迹五参数

- 全局最小二乘法
- 卡曼滤波

带电径迹五参数

- d_0: 螺旋线于x-y平面投影与IP的距离
- ϕ_0: x-y平面内，POCA的方位角
- κ: 横动量的倒数（1/Pt），其符号反映粒子电荷的正负
- z_0: POCA的z坐标
- $tan\lambda$: λ为螺旋线与x-y平面的夹角

- IP (Initial Point): 正负电子对撞点
- POCA (Point of closest approach)

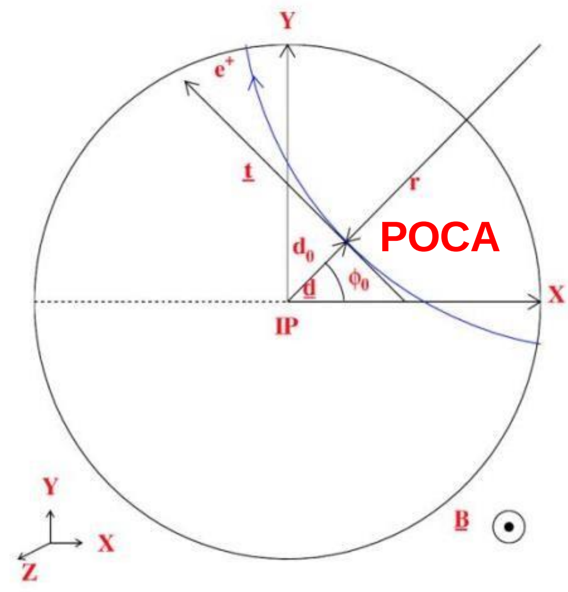
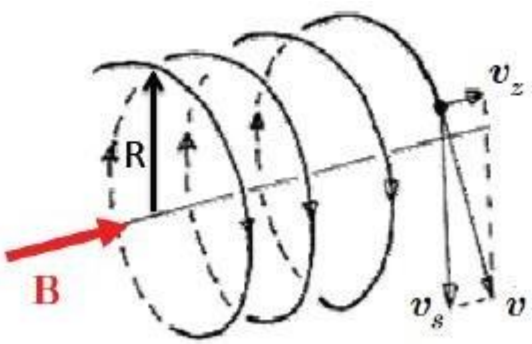
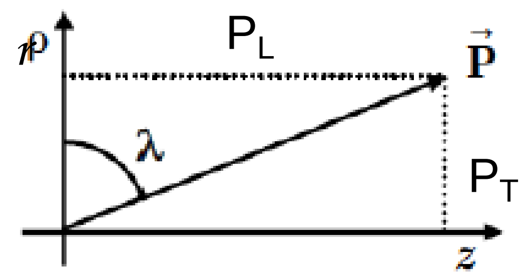


图 3.3 螺旋线径迹参数示意图



$$\mathbf{P} \equiv (d_0, \phi_0, \kappa, z_0, tan\lambda)^T.$$

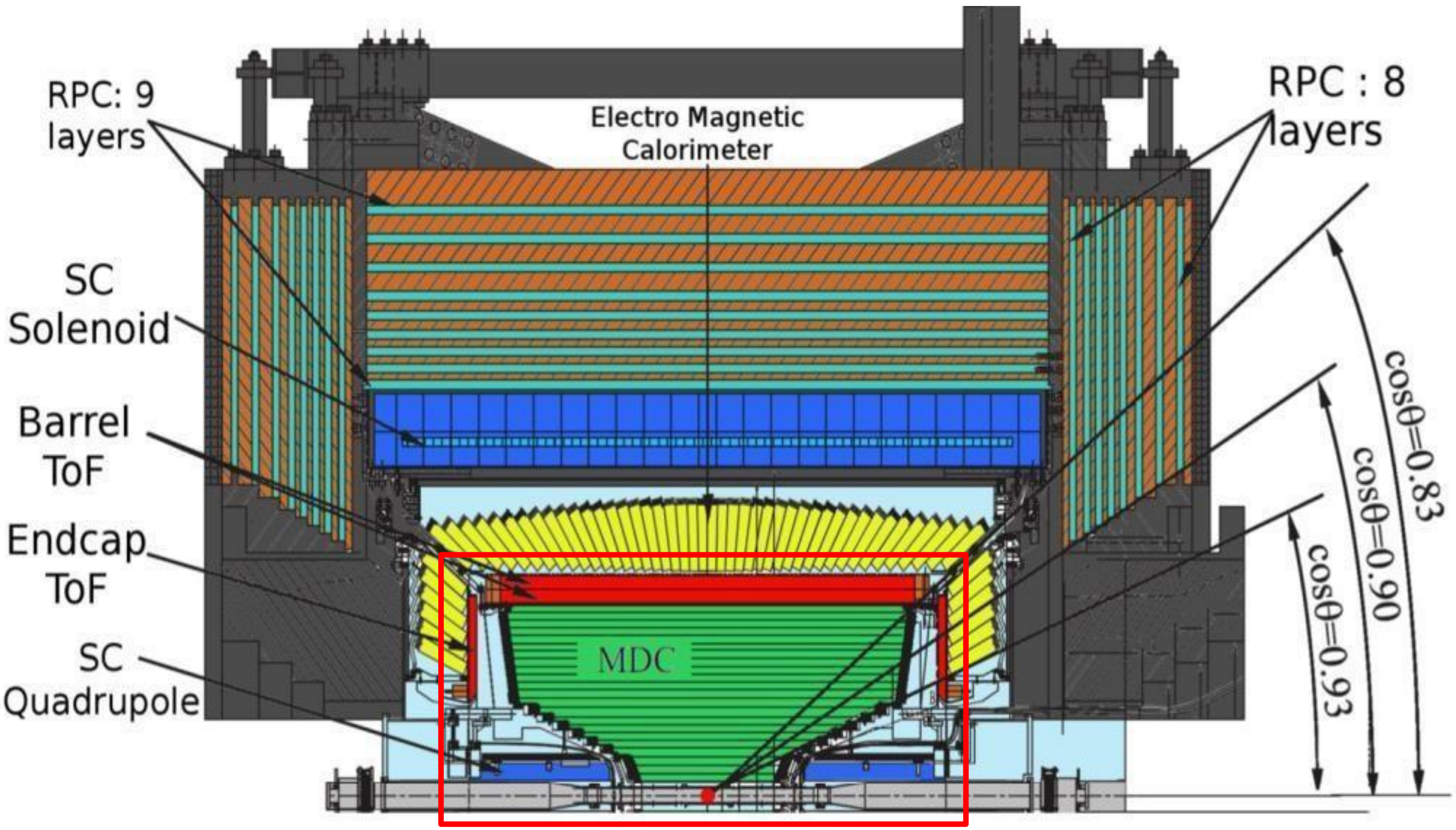


粒子鉴别系统

中国科学技术大学 University of Science and Technology of China

- MDC上的dEdx
- TOF上的飞行时间

（PID系统还包含EMC与MUC）



电离能损(dEdx)

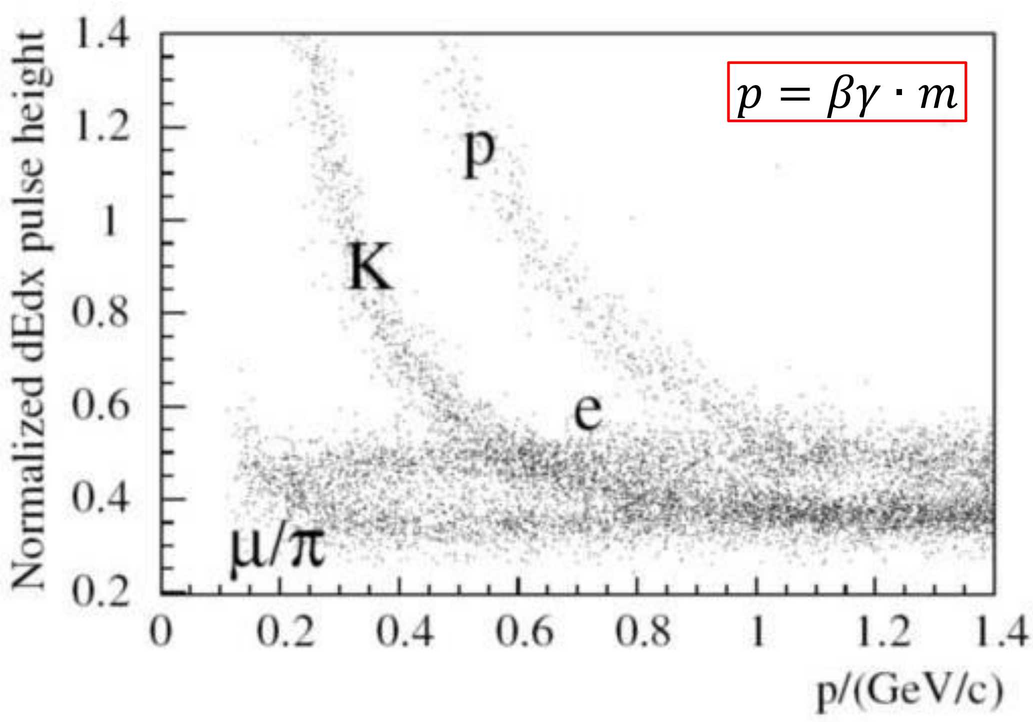
带电粒子与介质中的原子发生相互作用，损失的能量由Bethe-Bloch公式描述：

$$-\left\langle \frac{dE}{dx} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \ln \frac{2 m_e c^2 \beta^2 \gamma^2 T_{\max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right]$$

低动量区：$-\frac{dE}{dx} \propto \frac{1}{\beta^2}$，由漂移室得到粒子动量，再根据dEdx信息可实现粒子鉴别。

$$\chi = \frac{dE/dx(\text{测量}) - dE/dx(\text{期望})}{\sigma(dE/dx)}$$

不同粒子假设得到不同的期望值，与测量值比较可以得出被测粒子是某种粒子的几率



飞行时间探测器(TOF)

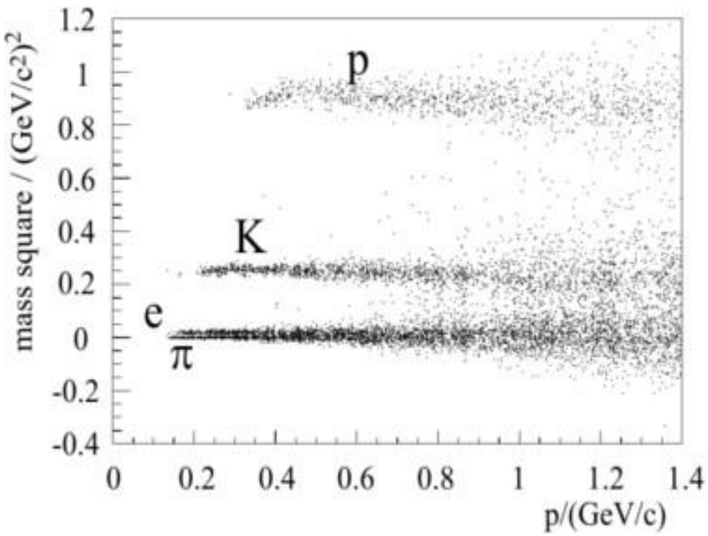
中国科学技术大学 University of Science and Technology of China

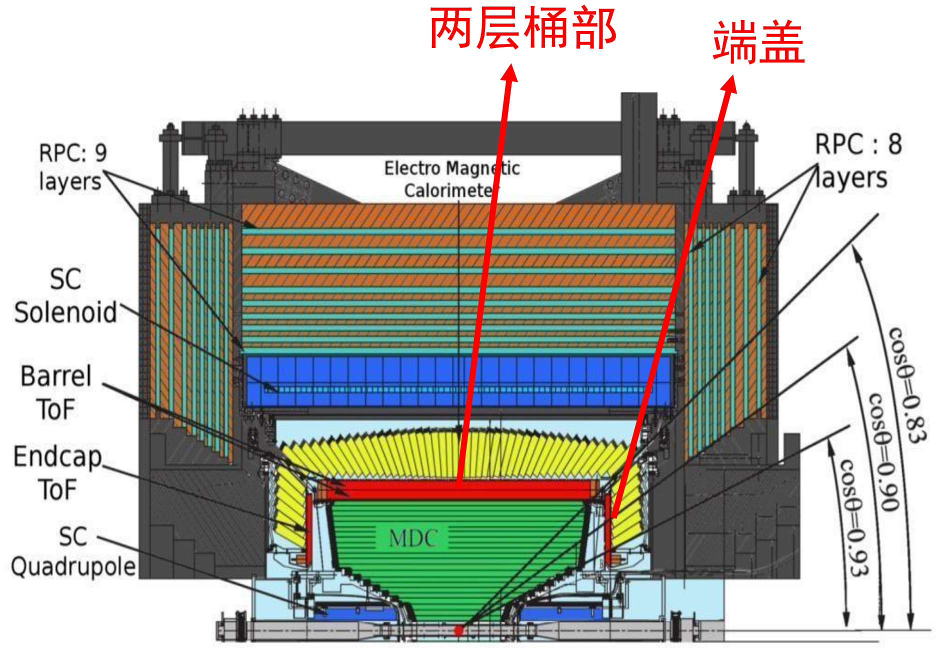
与dEdx类似，通过探测器测量得到的粒子动量p和飞行距离L，在粒子假设下，可计算得到预期飞行时间：

$$T_i^{Predict} = \frac{L}{c\beta_i}, \qquad \beta_i = p/E_i, \qquad E_i = \sqrt{m_i^2 + p^2}, \qquad (i = e, \mu, \pi, K, p)$$

与测量时间比较：

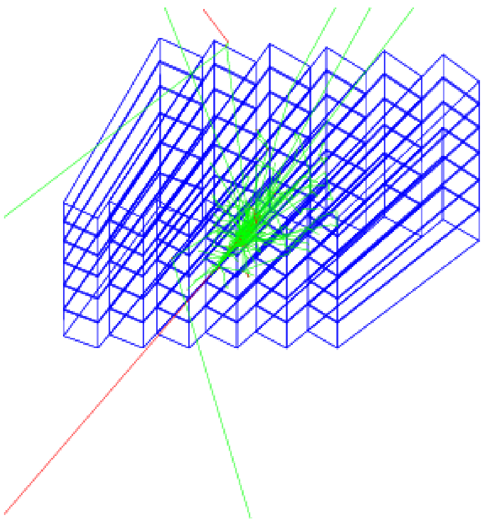
$$\chi^2 = \left(\frac{T^{Measure} - T_i^{Predict}}{\sigma}\right)^2$$

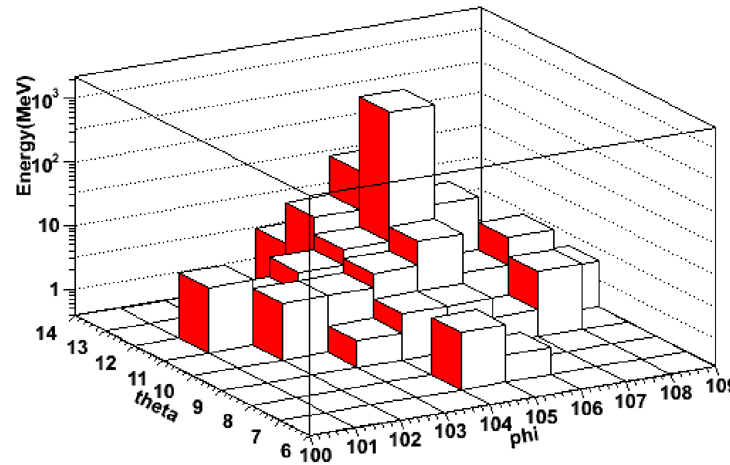




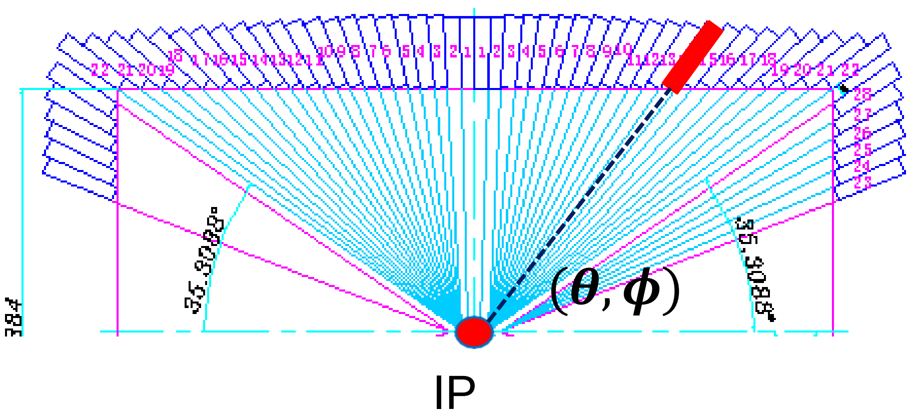
EMC探测光子能量

- 高能光子或正负电子于晶体中产生**电磁簇射**





- 光子四动量重建：

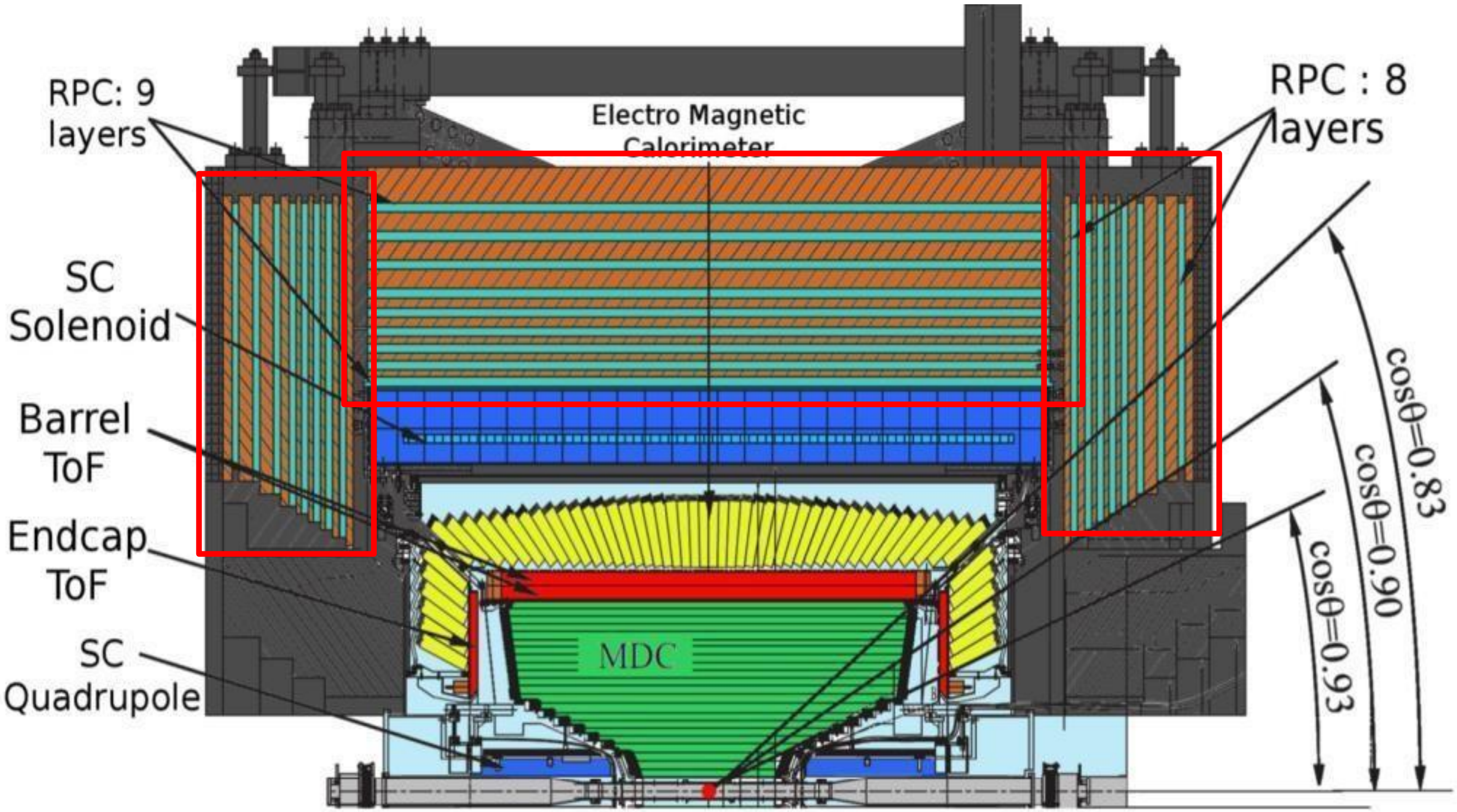


通过光子的位置信息得到其四动量：

$$P^{\mu} = (E, p)$$

MUC

- 测量末态μ/π的大概位置，与内层探测器的粒子径迹进行匹配
- 提高μ/π鉴别



参考材料

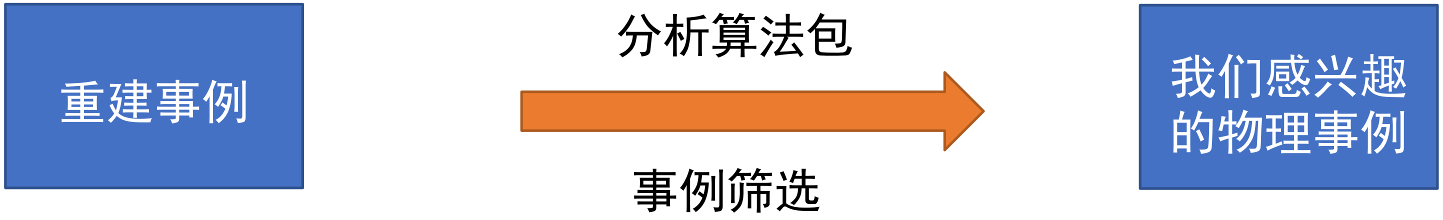
- Drift Chamber----Beijing Spectrometer(BEIII) Experiment (cas.cn)
- https://www.sciencedirect.com/science/article/pii/S0168900209023870
- 《北京谱仪(BESIII)的设计与研制》王贻芳著

二、Rhopi分析算法介绍

分析算法包**不能逐行理解**，需要理解每一块儿的内容

背景介绍

分析算法包用来做什么的？？？



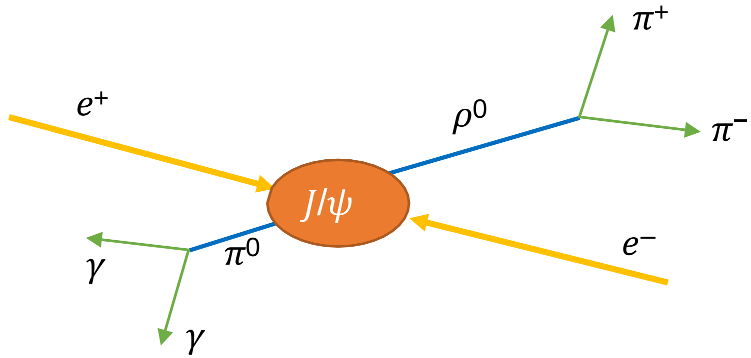
背景介绍

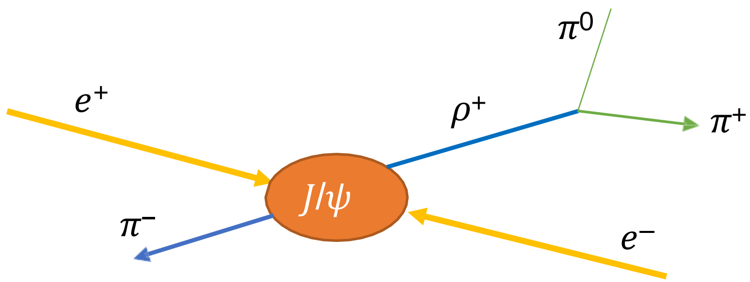
- 以rhopi算法包为例，级联衰变如下：

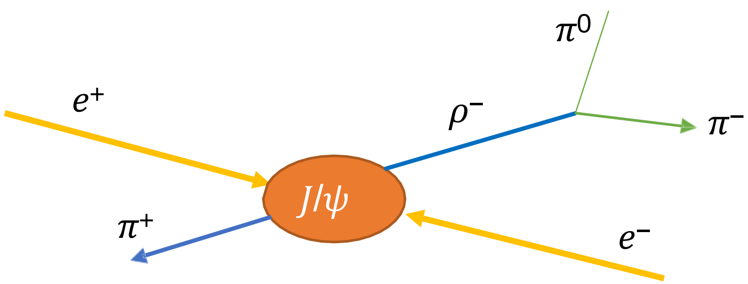
$$e^+e^- \rightarrow J/\psi$$

- $J/\psi \rightarrow \rho^0\pi^0,\ \rho^0 \rightarrow \pi^+\pi^-$
- $J/\psi \rightarrow \rho^+\pi^-,\ \rho^+ \rightarrow \pi^+\pi^0$
- $J/\psi \rightarrow \rho^-\pi^+,\ \rho^- \rightarrow \pi^-\pi^0$
- $\pi^0 \rightarrow \gamma\gamma$

末态粒子：$\pi^+\pi^-\gamma\gamma$ (2个带电径迹和两个光子)







算法包结构

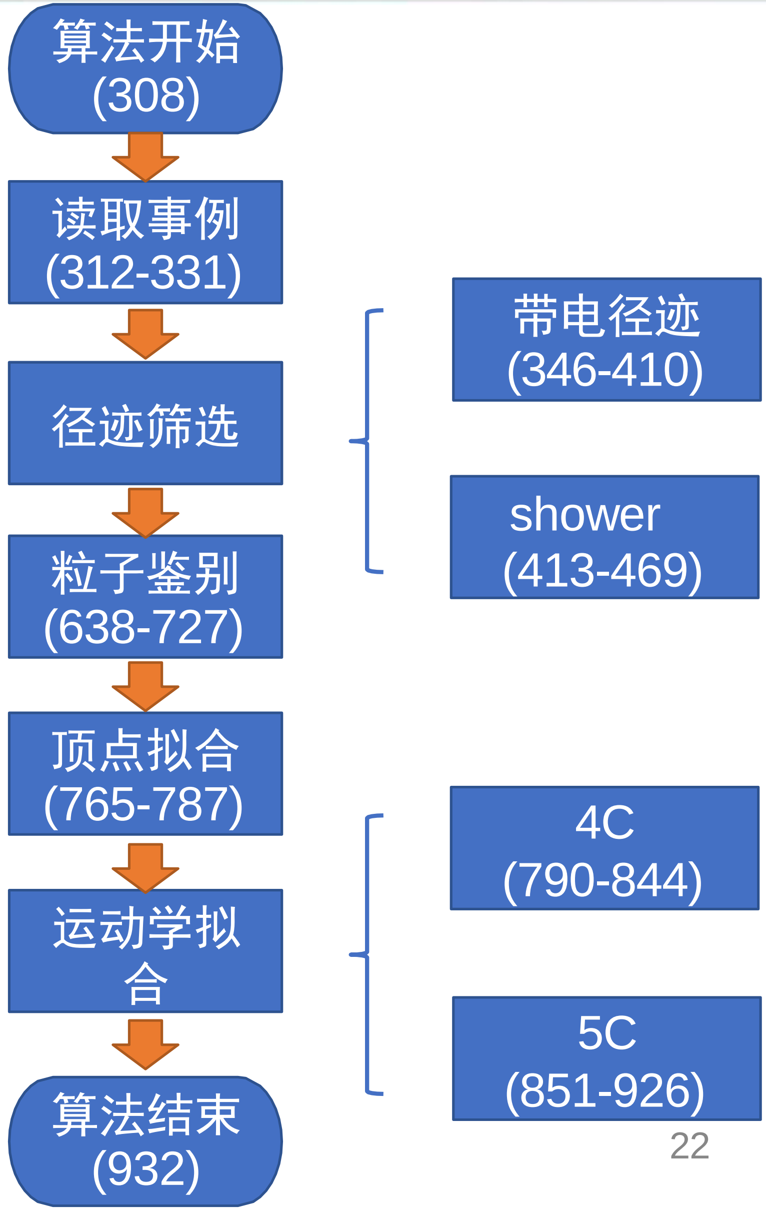
算法包路径：
/cvmfs/bes3.ihep.ac.cn/bes3sw/Boss/7.0.8/Analysis/Physics/RhopiAlg/RhopiAlg-00-00-23

```
[haokd@lxslc711 RhopiAlg-00-00-23]$ ls
cmt  CVS  RhopiAlg  share  src  x86_64-slc6-gcc46-opt
```

/src	源文件(*.cxx) (就是你的分析程序)
/<package-name>	头文件(*.h) (里面包含程序中使用到的变量)
/cmt	编译的文件(cmt clean→cmt config→cmt make)
/share	jobOptions文件
/x86_bulabula	编译算法包后产生的

程序框架(*.cxx)

- initialize()：初始化
- execute()： 算法开始
- finalize()： 算法结束



“开关”

- “开关”可以从ana.txt直接更改
- 避免多次编译
- 分析程序中设置default值，外部修改则以外部为主。

src/rhopi.cxx

```
Rhopi::Rhopi(const std::string& name, ISvcLocator* pSvcLocator) :
  Algorithm(name, pSvcLocator) {

  //Declare the properties
  declareProperty("Vr0cut", m_vr0cut=1.0);
  declareProperty("Vz0cut", m_vz0cut=5.0);
  declareProperty("EnergyThreshold", m_energyThreshold=0.04);
  declareProperty("GammaPhiCut", m_gammaPhiCut=20.0);
  declareProperty("GammaThetaCut", m_gammaThetaCut=20.0);
  declareProperty("GammaAngleCut", m_gammaAngleCut=20.0);
  declareProperty("Test4C", m_test4C = 1);
  declareProperty("Test5C", m_test5C = 1);
  declareProperty("CheckDedx", m_checkDedx = 1);
  declareProperty("CheckTof",  m_checkTof = 1);
}
```

ana.txt (用于运行分析程序的脚本)

```
#include "$ROOTIOROOT/share/jobOptions_ReadRec.txt"
#include "$VERTEXFITROOT/share/jobOptions_VertexDbSvc.txt"
#include "$MAGNETICFIELDROOT/share/MagneticField.txt"
#include "$ABSCORROOT/share/jobOptions_AbsCor.txt"
#include "$MCDECAYMODESVCROOT/share/GetDecayMode.txt"
#include "$MEASUREDECMSSVCROOT/share/anaOptions.txt"
#include "/workfs2/bes/peiyp/workarea/7.0.5/Analysis/Physics/RhopiAlg/RhopiAlg-00-00-23/share/jobOptions_Rhopi.txt"

Rhopi.GammaAngleCut = 15;
Rhopi.EnergyThreshold = 0.025;
Rhopi.Test4C=1;

// Input REC or DST file name
EventCnvSvc.digiRootInputFile ={

"/besfs4/offline/data/705-1/jpsi/round11/mc/tag/tag1/jpsi2018_stream001_run52962_file1.dst"

};

// Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )
MessageSvc.OutputLevel = 5;

// Number of events to be processed (default is 10)
ApplicationMgr.EvtMax = 10000;

ApplicationMgr.HistogramPersistency = "ROOT";
NTupleSvc.Output = { "FILE1 DATAFILE='res1.root' OPT='NEW'TYP='ROOT'"};
```

Initialize()

中国科学技术大学
University of Science and Technology of China

rhopi.cxx

```
StatusCode Rhopi::initialize(){
  MsgStream log(msgSvc(), name());

  log << MSG::INFO << "in initialize()" << endmsg;

  StatusCode status;
  NTuplePtr nt1(ntupleSvc(), "FILE1/vxyz");
  if ( nt1 ) m_tuple1 = nt1;
  else {
    m_tuple1 = ntupleSvc()->book ("FILE1/vxyz", CLID_ColumnWiseTuple, "ks N-Tuple example");
    if ( m_tuple1 )    {
      status = m_tuple1->addItem ("vx0",   m_vx0);
      status = m_tuple1->addItem ("vy0",   m_vy0);
      status = m_tuple1->addItem ("vz0",   m_vz0);
      status = m_tuple1->addItem ("vr0",   m_vr0);
      status = m_tuple1->addItem ("rvxy0",  m_rvxy0);
      status = m_tuple1->addItem ("rvz0",   m_rvz0);
      status = m_tuple1->addItem ("rvphi0", m_rvphi0);
    }
    else    {
      log << MSG::ERROR << "    Cannot book N-tuple:" << long(m_tuple1) << endmsg;
      return StatusCode::FAILURE;
    }
  }
```

rhopi.h

```
NTuple::Tuple*  m_tuple1;
NTuple::Item<double>  m_vx0;
NTuple::Item<double>  m_vy0;
NTuple::Item<double>  m_vz0;
NTuple::Item<double>  m_vr0;
NTuple::Item<double>  m_rvxy0;
NTuple::Item<double>  m_rvz0;
NTuple::Item<double>  m_rvphi0;
```

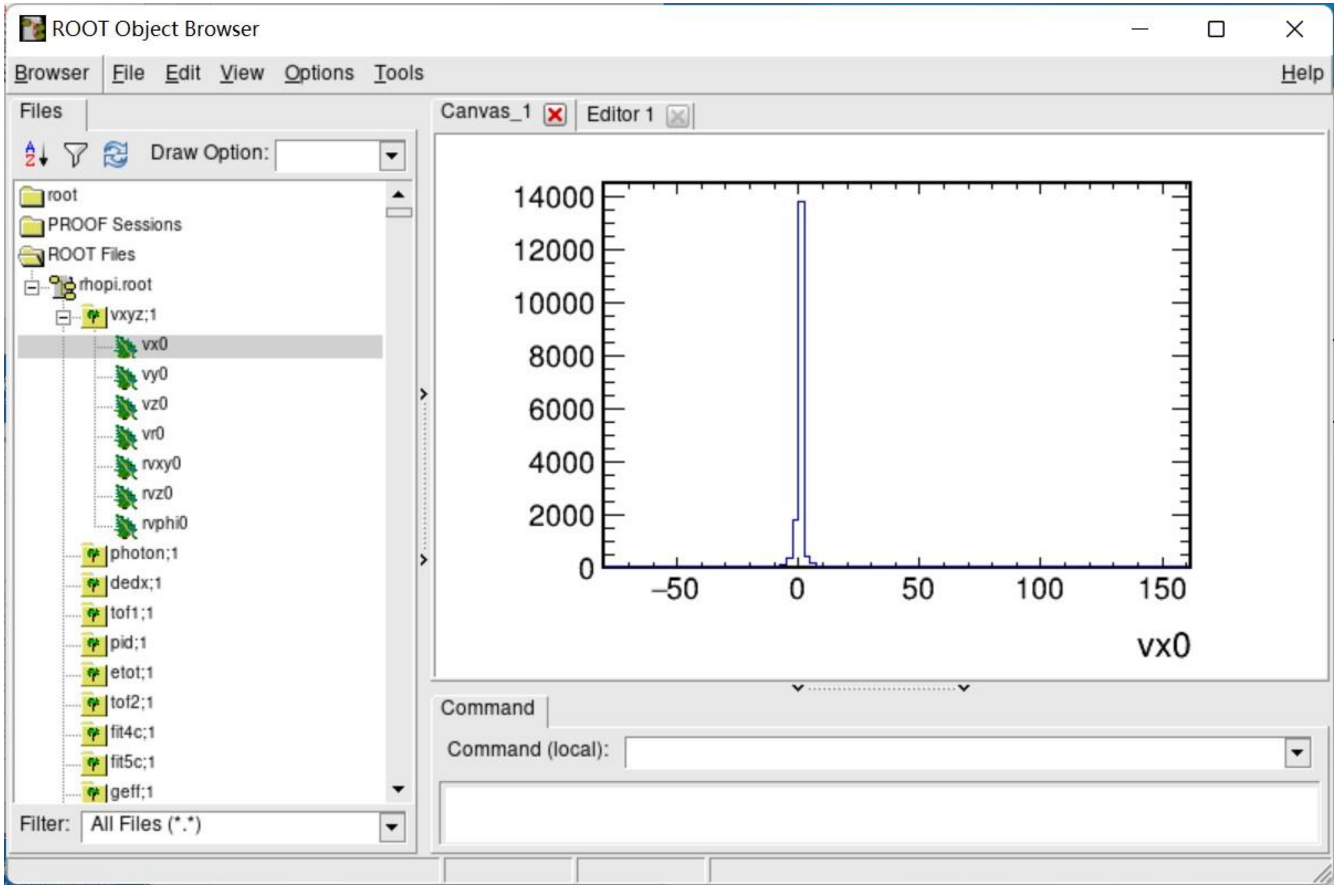
```
[peiyp@lxslc708 test]$ rl rhopi.root
root [0]
Attaching file rhopi.root as _file0...
root [1] .ls
TFile**         rhopi.root      Gaudi Trees
 TFile*         rhopi.root      Gaudi Trees
  KEY: TTree    vxyz;1  ks N-Tuple example
  KEY: TTree    photon;1        ks N-Tuple example
  KEY: TTree    dedx;1  ks N-Tuple example
  KEY: TTree    tof1;1  ks N-Tuple example
  KEY: TTree    pid;1   ks N-Tuple example
  KEY: TTree    etot;1  ks N-Tuple example
  KEY: TTree    tof2;1  ks N-Tuple example
  KEY: TTree    fit4c;1 ks N-Tuple example
  KEY: TTree    fit5c;1 ks N-Tuple example
  KEY: TTree    geff;1  ks N-Tuple example
  KEY: TTree    tofe;1  ks N-Tuple example
```

```
root [3] vxyz->Print()
******************************************************************************
*Tree    :vxyz      : ks N-Tuple example                                     *
*Entries :    17277 : Total =          973718 bytes  File  Size =     936003 *
*        :          : Tree compression factor =   1.04                       *
******************************************************************************
*Br    0 :vx0       : vx0/D                                                  *
*Entries :    17277 : Total  Size=     139040 bytes  File Size  =     133683 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.04     *
*............................................................................*
*Br    1 :vy0       : vy0/D                                                  *
*Entries :    17277 : Total  Size=     139040 bytes  File Size  =     133853 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.04     *
*............................................................................*
*Br    2 :vz0       : vz0/D                                                  *
*Entries :    17277 : Total  Size=     139040 bytes  File Size  =     134413 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.03     *
*............................................................................*
*Br    3 :vr0       : vr0/D                                                  *
*Entries :    17277 : Total  Size=     139040 bytes  File Size  =     134160 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.03     *
*............................................................................*
*Br    4 :rvxy0     : rvxy0/D                                                *
*Entries :    17277 : Total  Size=     139058 bytes  File Size  =     132724 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.04     *
*............................................................................*
*Br    5 :rvz0      : rvz0/D                                                 *
*Entries :    17277 : Total  Size=     139049 bytes  File Size  =     134401 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.03     *
*............................................................................*
*Br    6 :rvphi0    : rvphi0/D                                               *
*Entries :    17277 : Total  Size=     139067 bytes  File Size  =     131752 *
*Baskets :        5 : Basket Size=      32000 bytes  Compression=   1.05     *
*............................................................................*
```

- “m_”开头的变量需要在.h中声明，是程序中变量的名字
- 红色引号中的变量是外部root中的名字

Initialize()

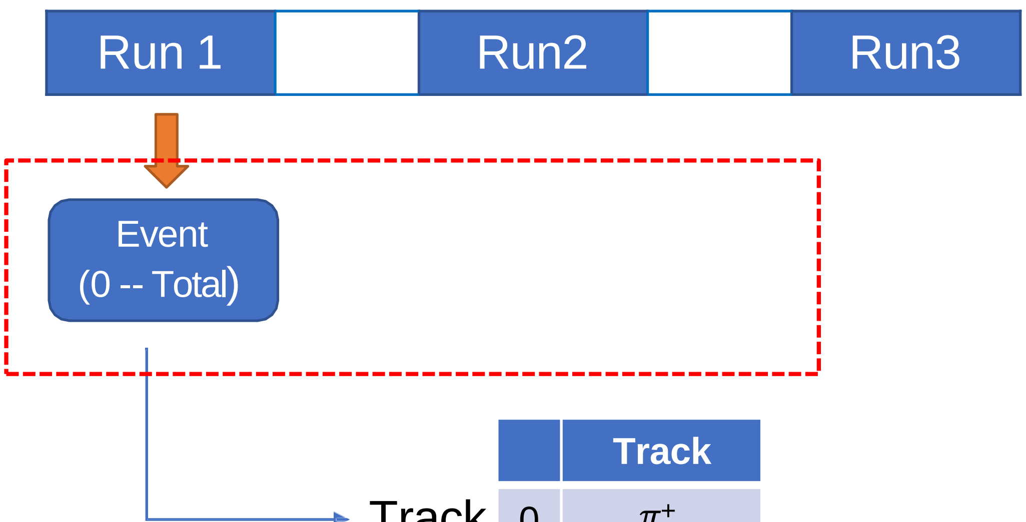
- 我们构建的数据结构为Tree，每一个变量都是一个histogram
- 打开方式：
  - root *.root
  - TBrowser –a（root 5下Tbrowser打不开)



execute()


```cpp
StatusCode Rhopi::execute() {

  std::cout << "execute()" << std::endl;

  MsgStream log(msgSvc(), name());
  log << MSG::INFO << "in execute()" << endreq;

  SmartDataPtr<Event::EventHeader> eventHeader(eventSvc(),"/Event/EventHeader");
  int runNo=eventHeader->runNumber();
  int event=eventHeader->eventNumber();
  log << MSG::DEBUG <<"run, evtnum = "
      << runNo << " , "
      << event <<endreq;
  cout<<"event "<<event<<endl;
  Ncut0++;

  SmartDataPtr<EvtRecEvent> evtRecEvent(eventSvc(), EventModel::EvtRec::EvtRecEvent);
  //  log << MSG::INFO << "get event tag OK" << endreq;
    log << MSG::DEBUG <<"ncharg, nneu, tottks = "
      << evtRecEvent->totalCharged() << " , "
      << evtRecEvent->totalNeutral() << " , "
      << evtRecEvent->totalTracks() <<endreq;
```

- 每个事例有唯一的run号与event号
- 分析程序每次execute()仅处理一个event，处理完毕进入下一个event
- Event by Event !!!



	Track
0	π^+
1	π^-
2	γ
3	γ

RUN STATUS LIST

ID	run_number	start_time	end_time	total_event	status
51705	71150	2022-01-19 08:33:45	2022-01-19 09:35:02	5058422	Good
51704	71149	2022-01-19 07:23:56	2022-01-19 08:25:22	4851754	Good
51703	71148	2022-01-19 06:26:00	2022-01-19 07:15:42	4192731	Good
51702	71147	2022-01-19 05:03:42	2022-01-19 06:05:47	4925395	Good
51701	71146	2022-01-19 03:54:26	2022-01-19 04:55:44	4876583	Good
51700	71145	2022-01-19 02:43:56	2022-01-19 03:45:46	4976439	Good
51699	71144	2022-01-19 01:34:22	2022-01-19 02:35:42	4914805	Good
51698	71143	2022-01-19 00:24:50	2022-01-19 01:25:51	4905122	Good
51697	71142	2022-01-18 23:13:41	2022-01-19 00:13:55	4898783	Good
51696	71141	2022-01-18 22:03:33	2022-01-18 23:05:54	5020576	Good

Run Number : 71150 **Status :** Good

Start Time : 2022-01-19 08:33:45 **End Time :** 2022-01-19 09:35:02

Total number of Events : 5058422 **initialied beam energy :** 1.8932GeV

Magnet Current : 3369.5872A **Lum(Begin Run) :** 19646.742 **Lum(End Run) :** 19648.674

Run Task : psi(3770) **beam type :** collision

HIT_MAP : Normal

HV : Normal

Other Problems : NO

Other comment :

Shift Chief : Zhefei Tian 2022-01-19 09:38:30

带电径迹筛选

University of Science and Technology of China

```
  for(int i = 0; i < evtRecEvent->totalCharged(); i++){
    EvtRecTrackIterator itTrk=evtRecTrkCol->begin() + i;
    if(!(*itTrk)->isMdcTrackValid()) continue;
    RecMdcTrack *mdcTrk = (*itTrk)->mdcTrack();
    double pch=mdcTrk->p();
    double x0=mdcTrk->x();
    double y0=mdcTrk->y();
    double z0=mdcTrk->z();
    double phi0=mdcTrk->helix(1);
    double xv=xorigin.x();
    double yv=xorigin.y();
    double Rxy=(x0-xv)*cos(phi0)+(y0-yv)*sin(phi0);
    m_vx0 = x0;
    m_vy0 = y0;
    m_vz0 = z0;
    m_vr0 = Rxy;

    HepVector a = mdcTrk->helix();
    HepSymMatrix Ea = mdcTrk->err();
    HepPoint3D point0(0.,0.,0.);   // the initial point for MDC recosntruction
    HepPoint3D IP(xorigin[0],xorigin[1],xorigin[2]);
    VFHelix helixip(point0,a,Ea);
    helixip.pivot(IP);
    HepVector vecipa = helixip.a();
    //helix P = (d0,phi0,kappa,z0,tan(lambda))
    double  Rvxy0=fabs(vecipa[0]);  //the nearest distance to IP in xy plane
    double  Rvz0=vecipa[3];         //the nearest distance to IP in z direction
    double  Rvphi0=vecipa[1];
    m_rvxy0=Rvxy0;
    m_rvz0=Rvz0;
    m_rvphi0=Rvphi0;

    m_tuple1->write();
//    if(fabs(z0) >= m_vz0cut) continue;
//    if(fabs(Rxy) >= m_vr0cut) continue;

    if(fabs(Rvz0) >= 10.0) continue;
    if(fabs(Rvxy0) >= 1.0) continue;

    iGood.push_back(i);
    nCharge += mdcTrk->charge();
  }
  // Finish Good Charged Track Selection
  //
  int nGood = iGood.size();
  log << MSG::DEBUG << "ngood, totcharge = " << nGood << " , " << nCharge << endreq;
  if((nGood != 2)||(nCharge!=0)){
    return StatusCode::SUCCESS;
  }
```

```
    IVertexDbSvc*  vtxsvc;
   Gaudi::svcLocator()->service("VertexDbSvc", vtxsvc);
   if(vtxsvc->isVertexValid()){
   double* dbv = vtxsvc->PrimaryVertex();
   double*  vv = vtxsvc->SigmaPrimaryVertex();
//     HepVector dbv = m_reader.PrimaryVertex(runNo);
//     HepVector vv = m_reader.SigmaPrimaryVertex(runNo);
     xorigin.setX(dbv[0]);
     xorigin.setY(dbv[1]);
     xorigin.setZ(dbv[2]);
   }
```

- IP：初始顶点，从数据库读取。
- 径迹需要限制在一个小范围内（排除宇宙线与束流本底）
- 电荷守恒以及带电径迹数量要求

Return SUCCESS; 中断这个event

光子筛选

中国科学技术大学
University of Science and Technology of China

```
  Vint iGam;
  iGam.clear();
  for(int i = evtRecEvent->totalCharged(); i< evtRecEvent->totalTracks(); i++) {
    EvtRecTrackIterator itTrk=evtRecTrkCol->begin() + i;
    if(!(*itTrk)->isEmcShowerValid()) continue;
    RecEmcShower *emcTrk = (*itTrk)->emcShower();
    Hep3Vector emcpos(emcTrk->x(), emcTrk->y(), emcTrk->z());
    // find the nearest charged track
    double dthe = 200.;
    double dphi = 200.;
    double dang = 200.;
    for(int j = 0; j < evtRecEvent->totalCharged(); j++) {
      EvtRecTrackIterator jtTrk = evtRecTrkCol->begin() + j;
      if(!(*jtTrk)->isExtTrackValid()) continue;
      RecExtTrack *extTrk = (*jtTrk)->extTrack();
      if(extTrk->emcVolumeNumber() == -1) continue;
      Hep3Vector extpos = extTrk->emcPosition();
      //       double ctht = extpos.cosTheta(emcpos);
      double angd = extpos.angle(emcpos);
      double thed = extpos.theta() - emcpos.theta();
      double phid = extpos.deltaPhi(emcpos);
      thed = fmod(thed+CLHEP::twopi+CLHEP::twopi+pi, CLHEP::twopi) - CLHEP::pi;
      phid = fmod(phid+CLHEP::twopi+CLHEP::twopi+pi, CLHEP::twopi) - CLHEP::pi;
      if(angd < dang){
        dang = angd;
        dthe = thed;
        dphi = phid;
      }
    }
    if(dang>=200) continue;
    double eraw = emcTrk->energy();
    dthe = dthe * 180 / (CLHEP::pi);
    dphi = dphi * 180 / (CLHEP::pi);
    dang = dang * 180 / (CLHEP::pi);
    m_dthe = dthe;
    m_dphi = dphi;
    m_dang = dang;
    m_eraw = eraw;
    m_tuple2->write();
    if(eraw < m_energyThreshold) continue;
    if(fabs(dang) < m_gammaAngleCut) continue;

    iGam.push_back(i);
  }

   //
   // Finish Good Photon Selection
   //
   int nGam = iGam.size();

   log << MSG::DEBUG << "num Good Photon " << nGam
   if(nGam<2){
     return StatusCode::SUCCESS;
   }
   Ncut2++;
```

对于i光子，循环所有带电径迹，找到i光子与所有带电径迹的最小夹角。一般设置最小夹角大于某个阈值，用于排除韧致辐射产生的shower。

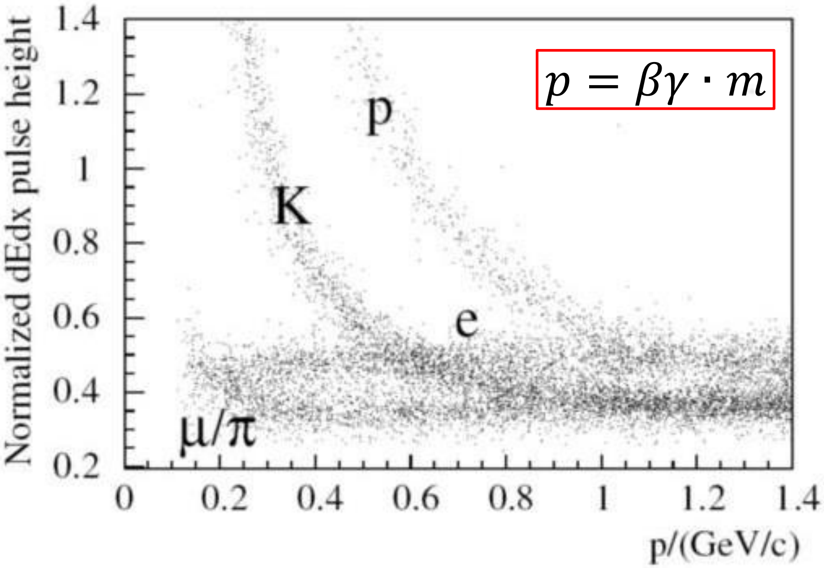
对于光子的要求通常大于等于n。因为EMC很“脏乱”，会有各种shower，需要后续进一步筛选。

```
  Vp4 pGam;
  pGam.clear();
  for(int i = 0; i < nGam; i++) {
    EvtRecTrackIterator itTrk = evtRecTrkCol->begin() + iGam[i];
    RecEmcShower* emcTrk = (*itTrk)->emcShower();
    double eraw = emcTrk->energy();
    double phi = emcTrk->phi();
    double the = emcTrk->theta();
    HepLorentzVector ptrk;
    ptrk.setPx(eraw*sin(the)*cos(phi));
    ptrk.setPy(eraw*sin(the)*sin(phi));
    ptrk.setPz(eraw*cos(the));
    ptrk.setE(eraw);

//     ptrk = ptrk.boost(-0.011,0,0);// boost to cms

    pGam.push_back(ptrk);
  }
```

光子四动量存储

检查dedx信息


```
  //
  // check dedx infomation
  //
  //

  if(m_checkDedx == 1) {
    for(int i = 0; i < nGood; i++) {
      EvtRecTrackIterator  itTrk = evtRecTrkCol->begin() + iGood[i];
      if(!(*itTrk)->isMdcTrackValid()) continue;
      if(!(*itTrk)->isMdcDedxValid())continue;
      RecMdcTrack* mdcTrk = (*itTrk)->mdcTrack();
      RecMdcDedx* dedxTrk = (*itTrk)->mdcDedx();
      m_ptrk = mdcTrk->p();

      m_chie = dedxTrk->chiE();
      m_chimu = dedxTrk->chiMu();
      m_chipi = dedxTrk->chiPi();
      m_chik = dedxTrk->chiK();
      m_chip = dedxTrk->chiP();
      m_ghit = dedxTrk->numGoodHits();
      m_thit = dedxTrk->numTotalHits();
      m_probPH = dedxTrk->probPH();
      m_normPH = dedxTrk->normPH();
      m_tuple7->write();
    }
  }
```



$$p = \beta\gamma \cdot m$$

测量带电粒子于MDC中的电离能损来鉴别粒子：

$$-\left\langle \frac{dE}{dx} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \ln \frac{2 m_e c^2 \beta^2 \gamma^2 T_{\max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right]$$

$$\chi = \frac{dE/dx(\text{测量}) - dE/dx(\text{期望})}{\sigma(dE/dx)}$$

低动量区： $-\frac{dE}{dx} \propto \frac{1}{\beta^2}$， 由漂移室得到粒子动量，再根据dEdx信息可实现粒子鉴别。

检查TOF信息

中国科学技术大学 University of Science and Technology of China

```
// check TOF infomation
//

if(m_checkTof == 1) {
  for(int i = 0; i < nGood; i++) {
    EvtRecTrackIterator  itTrk = evtRecTrkCol->begin() + iGood[i];
    if(!(*itTrk)->isMdcTrackValid()) continue;
    if(!(*itTrk)->isTofTrackValid()) continue;

    RecMdcTrack * mdcTrk = (*itTrk)->mdcTrack();
    SmartRefVector<RecTofTrack> tofTrkCol = (*itTrk)->tofTrack();

    double ptrk = mdcTrk->p();
    SmartRefVector<RecTofTrack>::iterator iter_tof = tofTrkCol.begin();
    for(;iter_tof != tofTrkCol.end(); iter_tof++ ) {
      TofHitStatus *status = new TofHitStatus;
      status->setStatus((*iter_tof)->status());
      if(!(status->is_barrel())){//endcap
        if( !(status->is_counter()) ) continue; // ?
        if( status->layer()!=0 ) continue;//layer1
        double path=(*iter_tof)->path(); // ?
        double tof  = (*iter_tof)->tof();
        double ph   = (*iter_tof)->ph();
        double rhit = (*iter_tof)->zrhit();
        double qual = 0.0 + (*iter_tof)->quality();
        double cntr = 0.0 + (*iter_tof)->tofID();
        double texp[5];
        for(int j = 0; j < 5; j++) {
          double gb = ptrk/xmass[j];
          double beta = gb/sqrt(1+gb*gb);
          texp[j] = 10 * path /beta/velc;
        }
        m_cntr_etof  = cntr;
        m_ptot_etof = ptrk;
        m_ph_etof   = ph;
        m_rhit_etof  = rhit;
        m_qual_etof  = qual;
        m_te_etof    = tof - texp[0];
        m_tmu_etof   = tof - texp[1];
        m_tpi_etof   = tof - texp[2];
        m_tk_etof    = tof - texp[3];
        m_tp_etof    = tof - texp[4];
        m_tuple8->write();
      }
      else {//barrel
        if( !(status->is_counter()) ) continue; // ?
        if(status->layer()==1){ //layer1
```

$$\chi = \frac{T(\text{测量}) - T(\text{期望})}{\sigma(T)}$$

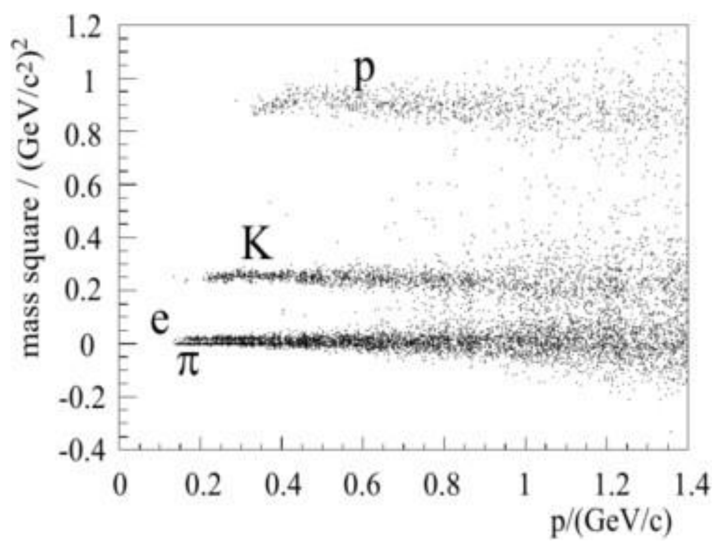
```
if(status->layer()==1){ //layer1

if(status->layer()==2){//layer2
```

TOF的桶部分为两层

- 测量带电粒子从IP点至TOF的飞行时间。

$$t = L/c\beta$$

- 与dE/dx鉴别类似，动量与飞行时间实现



粒子鉴别

中国科学技术大学
University of Science and Technology of China

```
  ParticleID *pid = ParticleID::instance();
  for(int i = 0; i < nGood; i++) {
    EvtRecTrackIterator itTrk = evtRecTrkCol->begin() + iGood[i];
    //    if(pid) delete pid;
    pid->init();
    pid->setMethod(pid->methodProbability());
//    pid->setMethod(pid->methodLikelihood());  //for Likelihood Method

    pid->setChiMinCut(4);
    pid->setRecTrack(*itTrk);
    pid->usePidSys(pid->useDedx() | pid->useTof1() | pid->useTof2() | pid->useTofE());
    pid->identify(pid->onlyPion() | pid->onlyKaon());    // seperater Pion/Kaon
    //    pid->identify(pid->onlyPion());
    //  pid->identify(pid->onlyKaon());
    pid->calculate();
    if(!(pid->IsPidInfoValid())) continue;
    RecMdcTrack* mdcTrk = (*itTrk)->mdcTrack();
    m_ptrk_pid = mdcTrk->p();
    m_cost_pid = cos(mdcTrk->theta());
    m_dedx_pid = pid->chiDedx(2);
    m_tof1_pid = pid->chiTof1(2);
    m_tof2_pid = pid->chiTof2(2);
    m_prob_pid = pid->probPion();
    m_tuple11->write();

    if(pid->probPion() < 0.001 || (pid->probPion() < pid->probKaon())) continue;
//    if(pid->probPion() < 0.001) continue;
//    if(pid->pdf(2)<pid->pdf(3)) continue; //  for Likelihood Method(0=electron 1=muo

    RecMdcKalTrack* mdcKalTrk = (*itTrk)->mdcKalTrack();//After ParticleID, use RecMdc
    RecMdcKalTrack::setPidType  (RecMdcKalTrack::pion);//PID can set to electron, muon

    if(mdcKalTrk->charge() >0) {
      ipip.push_back(iGood[i]);
      HepLorentzVector ptrk;
      ptrk.setPx(mdcKalTrk->px());
      ptrk.setPy(mdcKalTrk->py());
      ptrk.setPz(mdcKalTrk->pz());
      double p3 = ptrk.mag();
      ptrk.setE(sqrt(p3*p3+mpi*mpi));

//      ptrk = ptrk.boost(-0.011,0,0);//boost to cms

      ppip.push_back(ptrk);
    } else {
```

dEdx与TOF联合鉴别

```
RecMdcKalTrack *pipTrk = (*(evtRecTrkCol->begin()+ipip[0]))->mdcKalTrack();
RecMdcKalTrack *pimTrk = (*(evtRecTrkCol->begin()+ipim[0]))->mdcKalTrack();

WTrackParameter wvpipTrk, wvpimTrk;
wvpipTrk = WTrackParameter(mpi, pipTrk->getZHelix(), pipTrk->getZError());
wvpimTrk = WTrackParameter(mpi, pimTrk->getZHelix(), pimTrk->getZError());
```

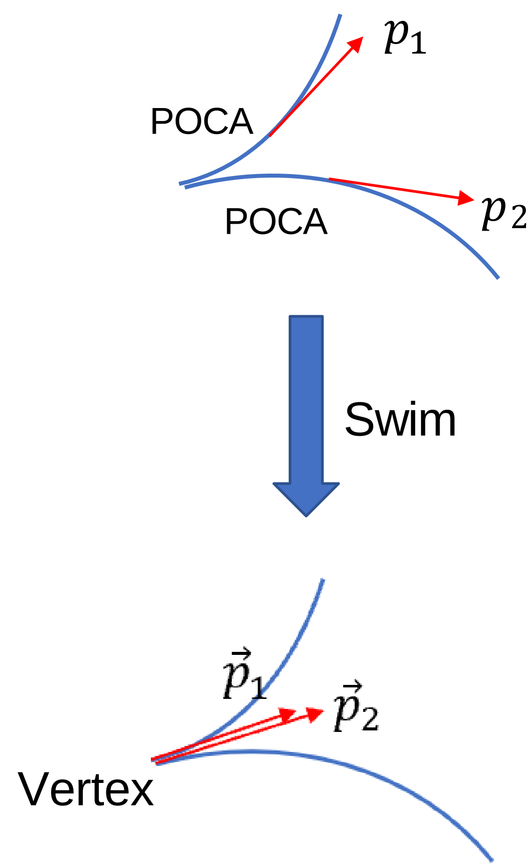
PID之后，已确定粒子类型，需赋予径迹质量。RecMdcTrack升级为RecMdcKalTrack。可拿到径迹参数，用于顶点拟合。

顶点拟合

```
//
//    Test vertex fit
//

HepPoint3D vx(0., 0., 0.);
HepSymMatrix Evx(3, 0);
double bx = 1E+6;
double by = 1E+6;
double bz = 1E+6;
Evx[0][0] = bx*bx;
Evx[1][1] = by*by;
Evx[2][2] = bz*bz;

VertexParameter vxpar;
vxpar.setVx(vx);
vxpar.setEvx(Evx);

VertexFit* vtxfit = VertexFit::instance();
vtxfit->init();
vtxfit->AddTrack(0,  wvpipTrk);
vtxfit->AddTrack(1,  wvpimTrk);
vtxfit->AddVertex(0, vxpar,0, 1);
if(!vtxfit->Fit(0)) return SUCCESS;
vtxfit->Swim(0);

WTrackParameter wpip = vtxfit->wtrk(0);
WTrackParameter wpim = vtxfit->wtrk(1);
```

初始设置一个误差无穷大的顶点

- 末态径迹需要来自于同一个顶点
- 多数粒子寿命很短，衰变顶点与产生顶点“重合”。



运动学拟合


```
if(m_test4C==1) {
  HepLorentzVector ecms(0.034,0,0,3.097);
  double chisq = 9999.;
  int ig1 = -1;
  int ig2 = -1;
  for(int i = 0; i < nGam-1; i++) {
    RecEmcShower *g1Trk = (*(evtRecTrkCol->begin()+iGam[i]))->emcShower();
    for(int j = i+1; j < nGam; j++) {
      RecEmcShower *g2Trk = (*(evtRecTrkCol->begin()+iGam[j]))->emcShower();
      kmfit->init();
      kmfit->AddTrack(0, wpip);
      kmfit->AddTrack(1, wpim);
      kmfit->AddTrack(2, 0.0, g1Trk);
      kmfit->AddTrack(3, 0.0, g2Trk);
      kmfit->AddFourMomentum(0, ecms);
      bool oksq = kmfit->Fit();
      if(oksq) {
        double chi2 = kmfit->chisq();
        if(chi2 < chisq) {
          chisq = chi2;
          ig1 = iGam[i];
          ig2 = iGam[j];
        }
      }
    }
  }

  if(chisq < 200) {
    RecEmcShower *g1Trk = (*(evtRecTrkCol->begin()+ig1))->emcShower();
    RecEmcShower *g2Trk = (*(evtRecTrkCol->begin()+ig2))->emcShower();
    kmfit->init();
    kmfit->AddTrack(0, wpip);
    kmfit->AddTrack(1, wpim);
    kmfit->AddTrack(2, 0.0, g1Trk);
    kmfit->AddTrack(3, 0.0, g2Trk);
    kmfit->AddFourMomentum(0, ecms);
    bool oksq = kmfit->Fit();
    if(oksq) {
      HepLorentzVector ppi0 = kmfit->pfit(2) + kmfit->pfit(3);
      m_mpi0 = ppi0.m();
      m_chi1 = kmfit->chisq();
      m_tuple4->write();
      Ncut4++;
    }
  }
}
```

4C

- 运动学拟合利用运动学上的守恒定律对事例加约束。通常：
  - ✓ 初末态四动量守恒（四个约束）
  - ✓ 中间共振态约束
- 运动学拟合通常用来选出最好的光子或光子组合，对径迹有一定修正。

```
  if(m_test5C==1) {
//    double ecms = 3.097;
    HepLorentzVector ecms(0.034,0,0,3.097);
    double chisq = 9999.;
    int ig1 = -1;
    int ig2 = -1;
    for(int i = 0; i < nGam-1; i++) {
      RecEmcShower *g1Trk = (*(evtRecTrkCol->begin()+iGam[i]))->emcShower();
      for(int j = i+1; j < nGam; j++) {
        RecEmcShower *g2Trk = (*(evtRecTrkCol->begin()+iGam[j]))->emcShower();
        kmfit->init();
        kmfit->AddTrack(0, wpip);
        kmfit->AddTrack(1, wpim);
        kmfit->AddTrack(2, 0.0, g1Trk);
        kmfit->AddTrack(3, 0.0, g2Trk);
        kmfit->AddResonance(0, 0.135, 2, 3);
        kmfit->AddFourMomentum(1, ecms);
        if(!kmfit->Fit(0)) continue;
        if(!kmfit->Fit(1)) continue;  //if(!kmfit->Fit()) continue;
        bool oksq = kmfit->Fit();
        if(oksq) {
          double chi2 = kmfit->chisq();
          if(chi2 < chisq) {
            chisq = chi2;
            ig1 = iGam[i];
            ig2 = iGam[j];
          }
        }
      }
    }
```

5C

通过置信度检验判断拟合情况，χ^2越小，拟合越好。

信息保存

- 于initialize()中定义tree与tuple
- 变量赋值后，需要通过write()将其写入.root中
- Write()后，tuple中的m_变量失效

```
NTuplePtr nt5(ntupleSvc(), "FILE1/fit5c");
if ( nt5 ) m_tuple5 = nt5;
else {
  m_tuple5 = ntupleSvc()->book ("FILE1/fit5c", CLID_ColumnWiseTuple, "ks N-Tuple example");
  if ( m_tuple5 )    {
    status = m_tuple5->addItem ("chi2",  m_chi2);
    status = m_tuple5->addItem ("mrh0",  m_mrh0);
    status = m_tuple5->addItem ("mrhp",  m_mrhp);
    status = m_tuple5->addItem ("mrhm",  m_mrhm);
  }
  else    {
  log << MSG::ERROR << "    Cannot book N-tuple:" << long(m_tuple5) << endmsg;
  return StatusCode::FAILURE;
  }
}
```

变量的类型于.h中声明

```
    m_chi2  = kmfit->chisq();
    m_mrh0  = prho0.m();
    m_mrhp  = prhop.m();
    m_mrhm  = prhom.m();
    double eg1 = (kmfit->pfit(2)).e();
    double eg2 = (kmfit->pfit(3)).e();
    double fcos = abs(eg1-eg2)/ppi0.rho();
    m_tuple5->write();
```

finalize()

```
StatusCode Rhopi::finalize() {
  cout<<"total number:         "<<Ncut0<<endl;
  cout<<"nGood==2, nCharge==0: "<<Ncut1<<endl;
  cout<<"nGam>=2:              "<<Ncut2<<endl;
  cout<<"Pass Pid:             "<<Ncut3<<endl;
  cout<<"Pass 4C:              "<<Ncut4<<endl;
  cout<<"Pass 5C:              "<<Ncut5<<endl;
  cout<<"J/psi->rho0 pi0:      "<<Ncut6<<endl;
  MsgStream log(msgSvc(), name());
  log << MSG::INFO << "in finalize()" << endmsg;
  return StatusCode::SUCCESS;
}
```

- dst中的所有事例遍历完后，执行finalize()
- 用于展示cut flow
- 也可作为分析程序运行成功的一个标志

```
  log << MSG::DEBUG << "ngood, totcharge = " << nGood << " , " << nCharge << endreq;
  if((nGood != 2)||(nCharge!=0)){
    return StatusCode::SUCCESS;
  }
  Ncut1++;
```

```
  log << MSG::DEBUG << "num Good Photon " << nGam  << " , " <<evtRecEvent->totalNeutral()<<endreq;
  if(nGam<2){
    return StatusCode::SUCCESS;
  }
  Ncut2++;
```

```
  int npip = ipip.size();
  int npim = ipim.size();
  if(npip*npim != 1) return SUCCESS;

  Ncut3++;
```

Selection Criteria	Absolute Efficiency (%)	Relative Efficiency (%)
Good Charged Tracks and PID	66.74	–
Vertex Fit	65.22	97.73
Good Shower Selection	57.68	88.45
Kinematic Fit	48.39	83.89
$\chi^2_{sec} < 15$ and $L/\sigma_L > 2.0$	42.88	88.61
$\chi^2_{corr} < 70$	35.17	82.01
$\lvert M_{\bar{p}\pi^+} - M_{\bar{\Lambda}}\rvert < 8MeV/c^2$	33.19	94.37
$M^{recoil}_{\bar{p}\pi^+} < 1.15GeV/c^2$	32.92	99.19
π^0 Asymmetry	28.33	86.05
$\theta_{\Lambda,Shower} > 10°$	24.13	85.19
$0.88 < M^{recoil}_{n} < 0.98(GeV/c^2)$	21.09	87.39

finalize()

一般跑分析作业运行成功时的标志：

```
ApplicationMgr       INFO Application Manager Stopped successfully
Total number:             20000
Nmc:                      20000
nGood>=2:                 19599
npbar>=1 && npip>=1:      14303
nGam>=1:                  13136
Vertex sel(ST):           12669
Npick(ST):                12669
nGood>=6:                 1302
np>=1 && npim>=1:         1195
Vertex sel(DT):           1134
nep>=1 && nem>=1:         374
ApplicationMgr       INFO Application Manager Finalized successfully
ApplicationMgr       INFO Application Manager Terminated successfully
```

总结

- BESIII 探测器
  - 各个子探测器(MDC, TOF, EMC, MUC)的主要功能
- 分析算法的主要框架：
  - initialize()
  - execute() —— event by event
  - Finalize()
- execute() 主要包含：
  - 带电径迹与shower读取
  - 粒子鉴别(PID)
  - 顶点拟合
  - 运动学拟合

谢谢！