High Multiplicity Clustering Using the Barrel Electromagnetic Calorimeter at STAR

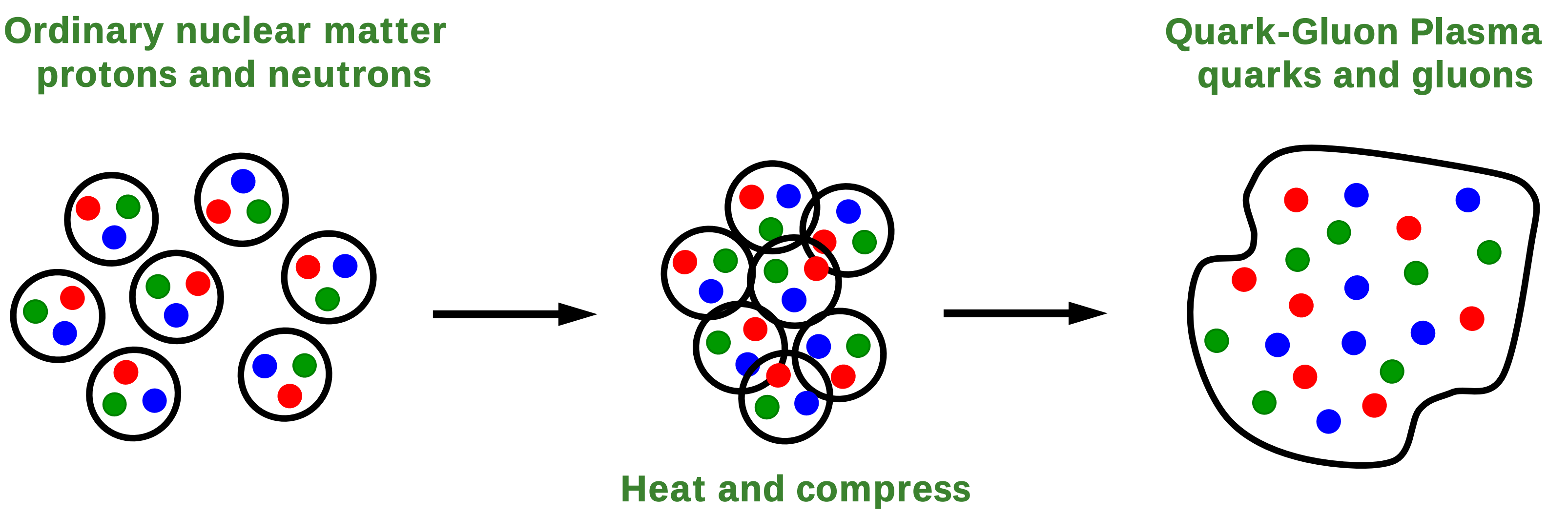
Darrick Jones – The College of New Jersey
Cyclotron Institute REU 2009
Advisor: Dr. Saskia Mioduszewski

Outline

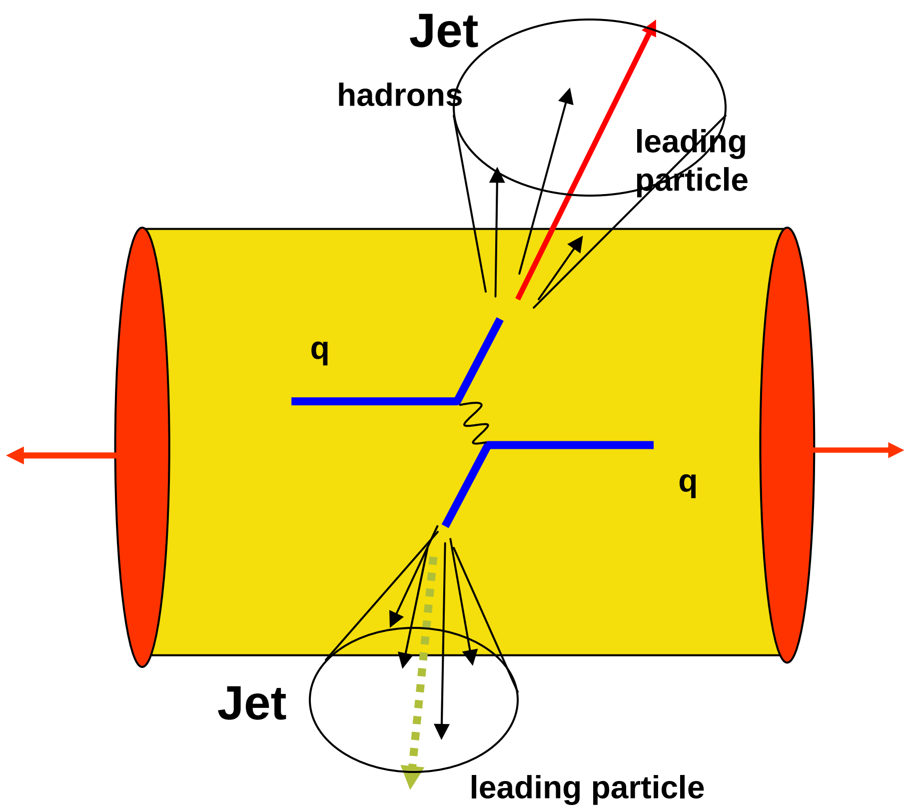
- Quark Gluon Plasma (QGP)
- Jets
- STAR – BEMC
- Clustering Algorithm
- Modularization

Quark Gluon Plasma

- Matter so hot (>10^{12} K) and dense that quarks and gluons (partons) are no longer confined to hadrons.
- Recreated in heavy ion collisions at RHIC.
- Believed to have existed up to 10^{-5} s after the Big Bang.
- Too short lived to be observed directly.
- Must be investigated indirectly.

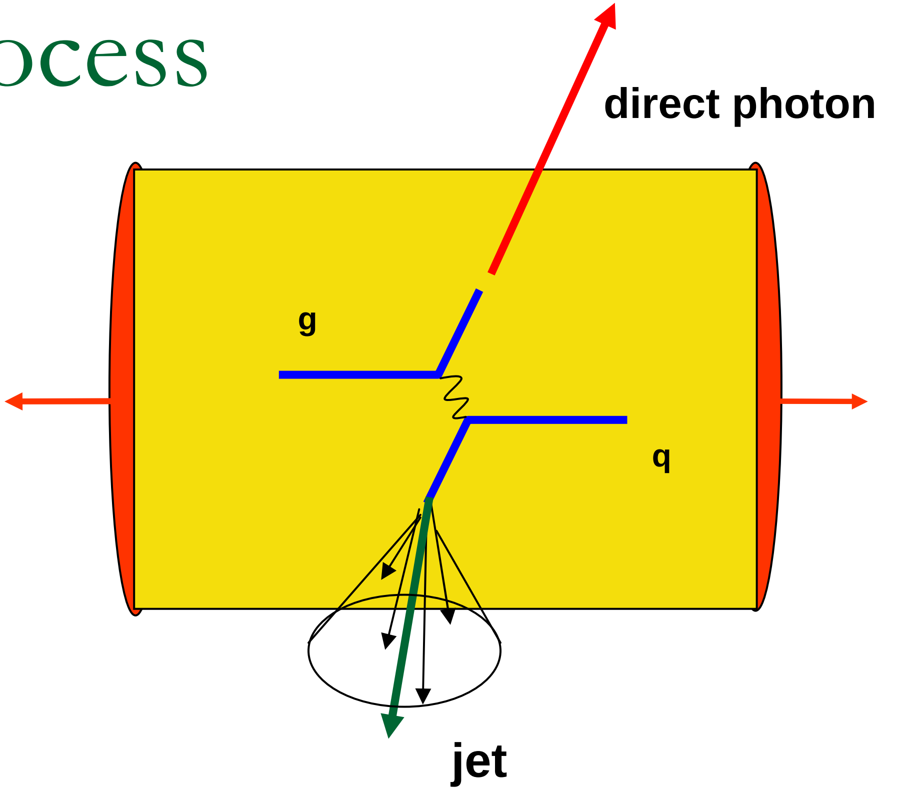


Hadron-Triggered Jets



- Large transfers of energy cause partons to scatter.
- Hard-scattered partons fragment into cones of hadrons forming back-to-back “jets”.
- Scattering occurs before matter is created.
- Hard-scattered partons traverse through and probe matter created.
- As they pass through the medium, they interact and lose energy.

γ-jet Process



- Jet produced back-to-back with “direct” photon.
- Photon maintains initial energy as it passes through the medium.
- Initial energy of parton and photon can be considered equal.
- By measuring energy loss of the jet, it can be determined how the parton was affected by the medium.

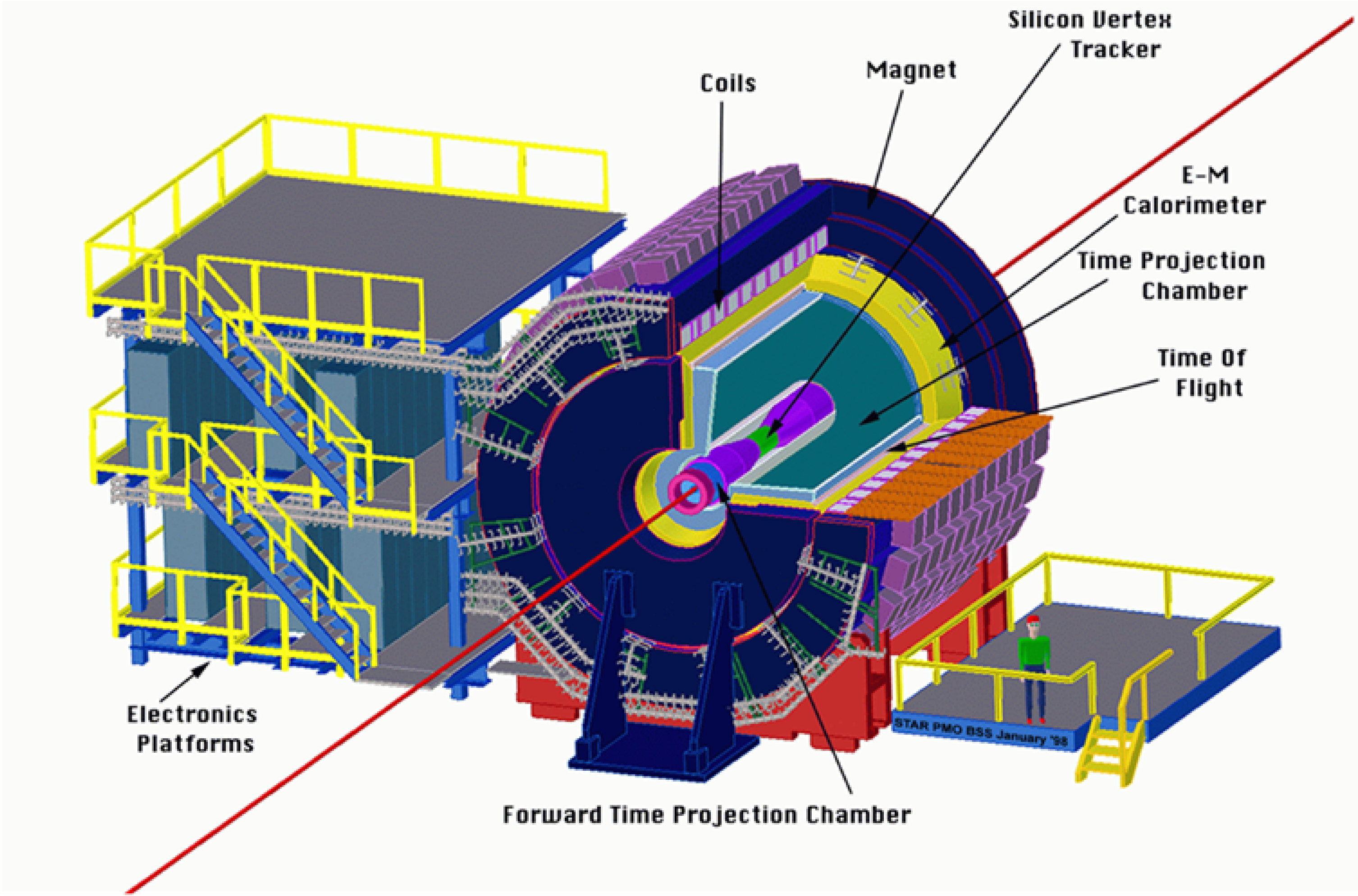
π^0 Decay

- Direct photons must be distinguished from background photons.
- Main source of background is π^0 decay.
- Primary decay mechanism:

$$\pi^0 \rightarrow \gamma + \gamma$$

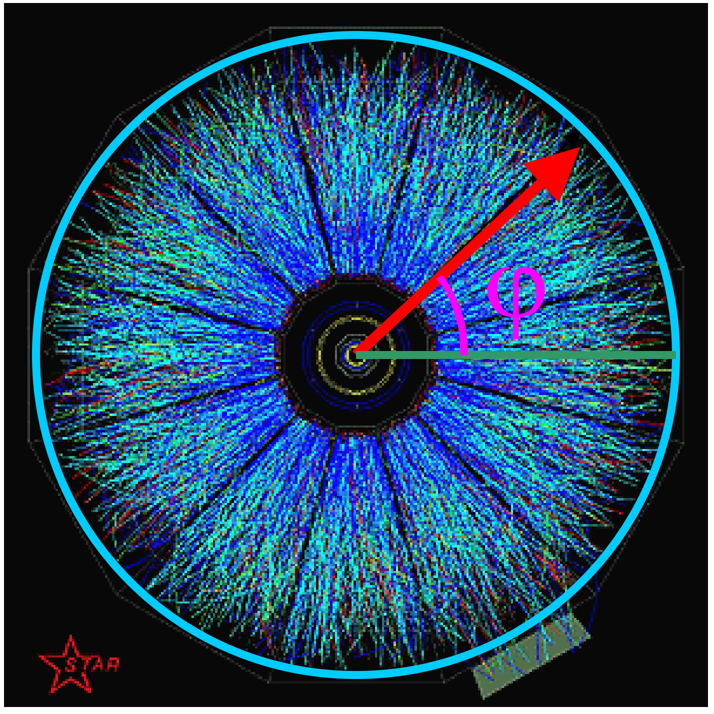
- Heavy ion collisions create an environment with lots of background.
- Need a clustering algorithm designed for high multiplicity.

Solenoid Tracker At RHIC (STAR)

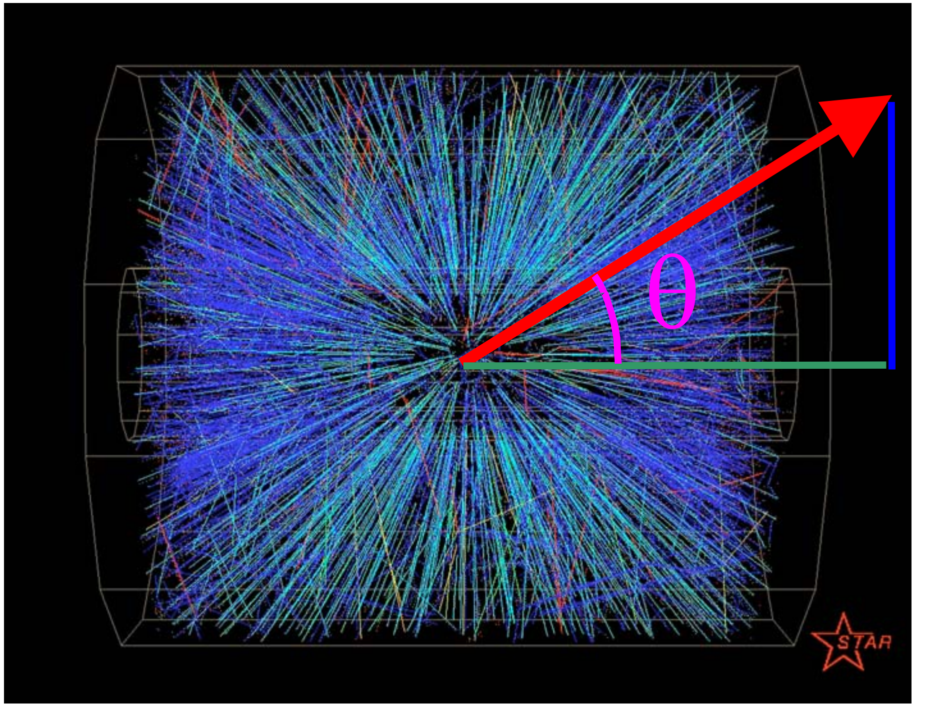


- STAR detector offers a unique myriad of subsystems which can be utilized to distinguish decay photons from direct photons.

Geometry of STAR



Beam View

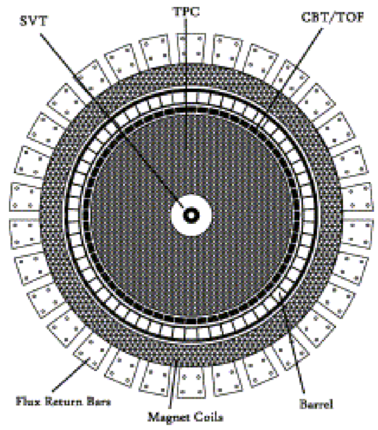


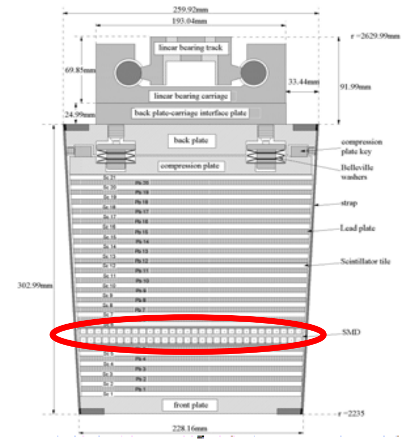
Side View

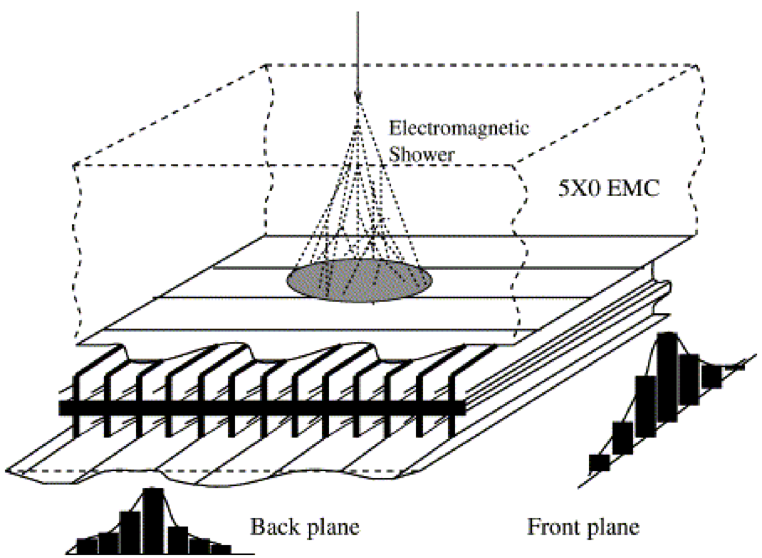
Pseudo-rapidity: $\eta = -\ln\left[\tan\left(\theta/2\right)\right]$

Barrel Electromagnetic Calorimeter (BEMC)

- Designed to record energy deposited from electromagnetic particles.
- 120 modules – 60 in φ by 2 in η.
- Each module is divided into 40 towers.
- Towers are 20 layers of lead and 21 layers of scintillator.



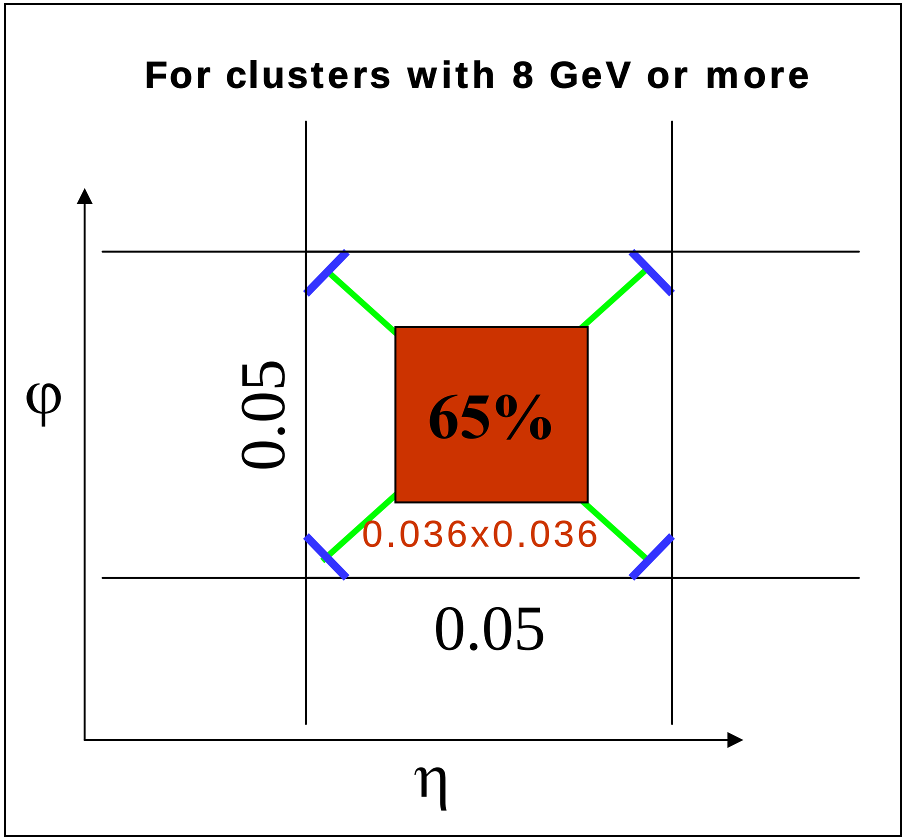




- Barrel Shower Max Detector (BSMD) is located after 5 layers of lead and scintillator.
- BSMD further divides each module into 150 strips in η and φ, 18000 subdivisions in each direction.
- High spatial resolution allows precise determination of particle position necessary for high multiplicity clustering.

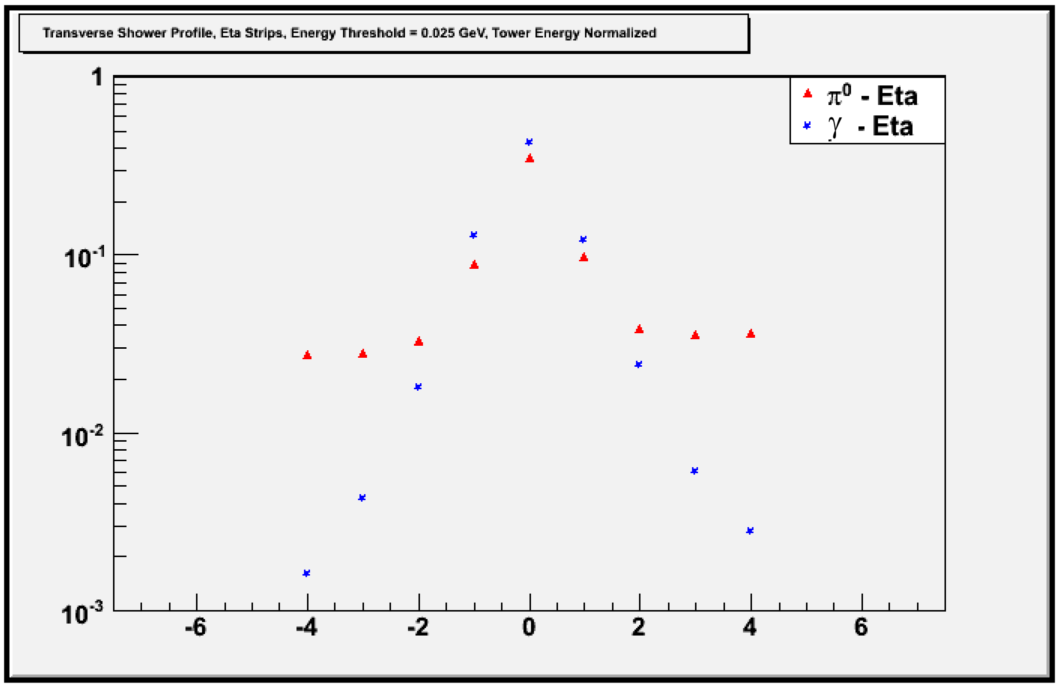
How the Algorithm Works

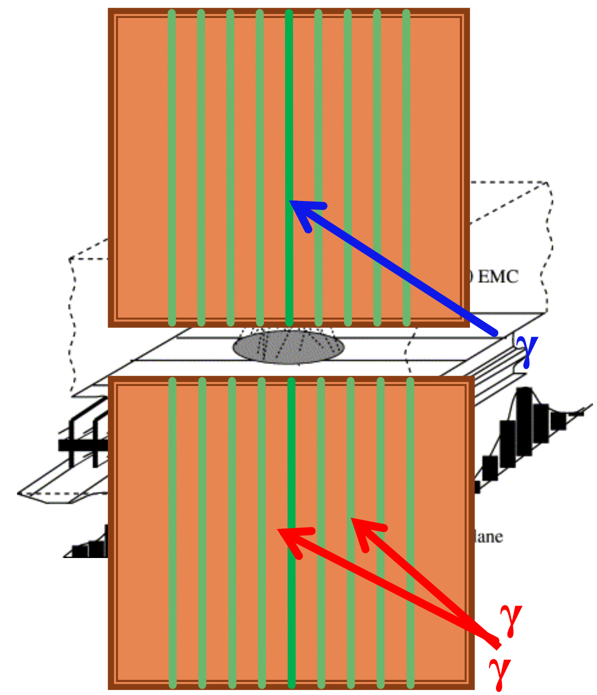
- Scans φ and η strips for peak energy above threshold.
- Identifies the tower which contains these strips.
- Checks total tower energy.
- Determines exact location of the hit using the BSMD.
- Finds 8 neighboring towers
- Compares position to find nearest neighbor.
- Based on location of hit, uses strip data from 1 or 2 towers



Shower Profile

- From the information gathered by the strips, a transverse shower profile is generated with high enough resolution that decay photons can be separated from direct photons (from simulated particles).





- Useful for rejecting background in the γ-jet Analysis

Code Modularization

- While the algorithm works very well, the program itself required substantial work.
- Program was not suitable for general use.
- The program needed to be made flexible, readable, and in compliance with STAR coding standards through modularization.

Improvements

- Able to make significant improvements using simple compu… as.
  - Header files I…
  - Structures of… cate variables

```
#include "StBemcUtilInfo.h"

struct BemcUtil //class designed to
ertain BEMC utility
{
   float             energy,
                     theta,
                     pedestal,
                     rms,
                     angle,
                     x,
                     y,
                     z,
                     gParentTrackP,
                     gParticleId;
   int               id,
                     tower,
                     sub,
                     status;
   unsigned  int     adc,
                     module,
                     eta;
   bool              central,
                     nextHiEtCheck,
                     nearestCheck,
                     nextHiEnergyCheck;
};
```

```
float Enpq1(0),Ennq1(0),Enpq20(0),          nnq21(0);
int towerpq1(0),towernq1(0),towerp          9(0),towerpq21(0),towernq21(0);
int Idpq1(0),Idnq1(0),Idpq20(0),Id          q21(0);
unsigned int adcpq1(0),adcnq1(0),a          0),adcpq21(0),adcnq21(0);
float thetapq1(0),thetanq1(0),thet          q19(0),thetapq21(0),thetanq21(0);
float ppq1(0),pnq1(0),ppq20(0),pnq          ;
float pidpq1(0),pidnq1(0),pidpq20          21(0),pidnq21(0);
UInt_t Modpq1(0),Modnq1(0),Modpq2          q21(0),Modnq21(0);
Int_t sUbpq1(0),sUbnq1(0),sUbpq20          21(0),sUbnq21(0);
UInt_t eTapq1(0),eTanq1(0),eTapq2          q21(0),eTanq21(0);

int nuOfNiq(0), nextIdq(0), nextI
float nextHiEtq(0),nextHiEq(0);
float nextPedq(0),nextRMSq(0),next          0); Int_t nextStq(0);
float pedpq1(0),pednq1(0),pedpq19          20(0),pednq20(0);
float RMSpq1(0), RMSnq1(0),RMSpq1          q20(0),RMSnq20(0);
Int_t spq1(0),snq1(0),spq19(0),sn          ;
```

Improvements

- Increase readability and functionality by using predefined functions available in STAR libraries.

Before

```
if(abs(didT-didTq)!=20&&abs(didT-didTq)!=19&&abs(didT-didTq)!=21&&abs(didT-didTq)!=1&&abs(didT-didTq)!=2380&&abs(didT-didTq)!=2381&&
abs(didT-didTq)!=2379&&eT>0&&eThq>0) continue;
if(abs(didT-didTq)!=20&&abs(didT-didTq)!=19&&abs(didT-didTq)!=21&&abs(didT-didTq)!=1&&abs(didT-didTq)!=2380&&abs(didT-didTq)!=2381&&
abs(didT-didTq)!=2379&&eT<0&&eThq<0) continue;

if(eT>0&&eThq<0&&fabs(fabs(eT)-fabs(eThq))!=0)continue; if(eT<0&&eThq>0&&fabs(fabs(eT)-fabs(eThq))!=0)continue;
if(fabs(fabs(eT)-fabs(eThq))>0.09 ||fabs(fabs(phT)-fabs(phiThq))>0.09) continue;
if(eT>0&&eThq<0&&fabs(fabs(eT)-fabs(eThq))==0&&eT>0.05&&eThq<-0.05)continue;
if(eT<0&&eThq>0&&fabs(fabs(eT)-fabs(eThq))==0&&eThq>0.05&&eT<-0.05)continue;
if(eT>0&&eThq<0&&fabs(fabs(eT)-fabs(eThq))==0&&eT<0.05&&eThq>-0.05&&phT>0&&phiThq<0)continue;//add here to these two conditions tower
if(eT>0&&eThq<0&&fabs(fabs(eT)-fabs(eThq))==0&&eT<0.05&&eThq>-0.05&&phT<0&&phiThq>0)continue;
if(eT<0&&eThq>0&&fabs(fabs(eT)-fabs(eThq))==0&&eT<0.05&&eThq>-0.05&&phT>0&&phiThq<0)continue;
if(eT<0&&eThq>0&&fabs(fabs(eT)-fabs(eThq))==0&&eT<0.05&&eThq>-0.05&&phT<0&&phiThq>0)continue;
```

After

```
StEmcPosition* mEmcPosition = new StEmcPosition();

for(a = -1; a < 2; a++)
  {
    for(b = -1; b < 2; b++)
      {
        if(a == 0 &&  b == 0) continue;
        tower[a+1][b+1].id = mEmcPosition->getNextTowerId(tower[1][1].id, b, a);
      }
  }

delete mEmcPosition;
```

Improvements

- Reduced redundancy by using loops and arrays.

```
if(didTq==didT+1&&phT==phiThq)
{Enpq1=Energyhq;Idpq1=didTq;adcpq
Modpq1=Modulehq;sUbpq1=Subhq;eTa

if((didTq==didT-1&&phT==phiThq)|
{Ennq1=Energyhq;Idnq1=didTq;adcn
Modnq1=Modulehq;sUbnq1=Subhq;eTa

if((didTq==didT+20||didTq==didT-
{Enpq20=Energyhq;Idpq20=didTq;ad
Modpq20=Modulehq;sUbpq20=Subhq;e

if((didTq==didT-20||didTq==didT+
{Ennq20=Energyhq;Idnq20=didTq;ad
 Modnq20=Modulehq;sUbnq20=Subhq;

if((didTq==didT+21||didTq==didT-
{Enpq21=Energyhq;Idpq21=didTq;ad
Modpq21=Modulehq;sUbpq21=Subhq;e

if(((didTq==didT-21||didTq==didT
||(fabs(fabs(phT)-fabs(phiThq))>
{Ennq21=Energyhq;Idnq21=didTq;ad
Modnq21=Modulehq;sUbnq21=Subhq;e

if(((didTq==didT+19||didTq==didT
||(fabs(fabs(phT)-fabs(phiThq))>
{Enpq19=Energyhq;Idpq19=didTq;ad
sUbpq19=Subhq;eTapq19=Etahq;pedp

if((didTq==didT-19||didTq==didT+
{Ennq19=Energyhq;Idnq19=didTq;ad
sUbnq19=Subhq;eTanq19=Etahq;pednq19=pedhhq;RMSnq19=RMShhq;snq19=sphhq;}
```

```
for(a = 0; a < 3; a++)
 {
   for(b = 0; b < 3; b++)
    {
      if(a == 1 && b == 1)continue;
      if(didTq == tower[a][b].id)
       {
         neighborCheck = 1;
       }
      if(neighborCheck == 1) break;
    }
   if(neighborCheck == 1) break;
 }

if(neighborCheck == 0)
  continue;

nuOfNiq++;

tower[a][b].energy        = hitsEwq[rwq]->energy();
tower[a][b].id            = didTq;
tower[a][b].adc           = hitsEwq[rwq]->adc();
tower[a][b].theta         = Thetahq;
tower[a][b].tower         = tower[1][1].id-tower[a][b].id;
tower[a][b].module        = Modulehq;
tower[a][b].sub           = Subhq;
tower[a][b].eta           = Etahq;
tower[a][b].pedestal      = mCalibPedValues[didTq-1];
tower[a][b].rms           = mCalibPedRMSValues[didTq-1];
tower[a][b].status        = mStatusT[didTq-1];
tower[a][b].gParentTrackP = ParentTrackP[tower[a][b].id-1];
tower[a][b].gParticleId   = ParticleId[tower[a][b].id-1];
```

Improvements

- By incorporating easily understandable variable names, as well as a structure, and adding member functions:
  - Gave the user the option to define threshold values by eliminating hard-coded values
  - Eliminated redundant loops.
  - Made code more readable.
  - Eliminated or replaced 200+ variables.
  - Decreased significantly the total number of statements.

Further Improvements

- Increase modularization by introducing functions which can be easily moved and replaced if needed.
- Add more flexibility by providing functions to set analysis-specific parameters
- More of the same.
- Ensure further compliance with STAR coding standards for eventual implementation into the STAR library.

Acknowledgements

- Saskia Mioduszewski
- Ahmed Hamed
- Sherry Yennello
- Larry May
- REU Group
- Texas A&M Cyclotron Institute
- NSF