φ resolution study by MC

4th report

Genki Nukazuka (RIKEN)

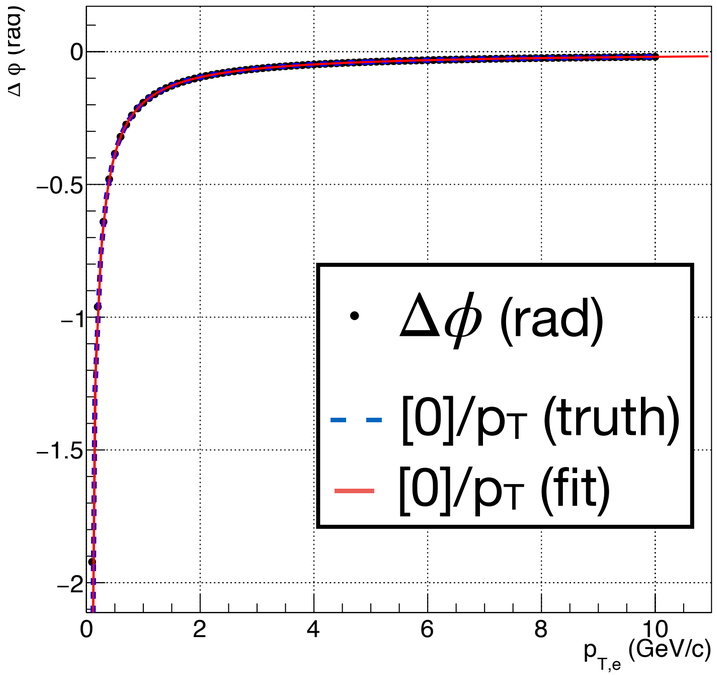
2nd report: Summary

Run	#event	Beam	Momentum (GeV/c)	ϕ dist (rad)	η dist	vertex (cm)	with EMCal hits asso. with beam Δϕ mean (mrad)	Δϕ std. dev. (mrad)	Δϕ resolution (%)
4	5.4E+04	μ^-	0.5	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-385.3	8.7	2.25%
1	5.0E+04	μ^-	1	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-186.0	3.9	2.07%
2	4.9E+04	μ^-	2	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-91.9	1.9	2.10%
3	5.0E+04	μ^-	4	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-45.7	2.1	4.54%
9	5.6E+04	e^-	0.5	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-392.1	12.3	3.14%
5	5.1E+04	e^-	1	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-189.4	5.8	3.07%
6	2.9E+04	e^-	2	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-94.3	3.5	3.76%
7	4.1E+04	e^-	4	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-50.6	4.7	9.27%
8	4.5E+04	e^-	8	[-π, π]	fixed at 0	fixed at (0, 0, 0)	-27.4	4.4	16.01%

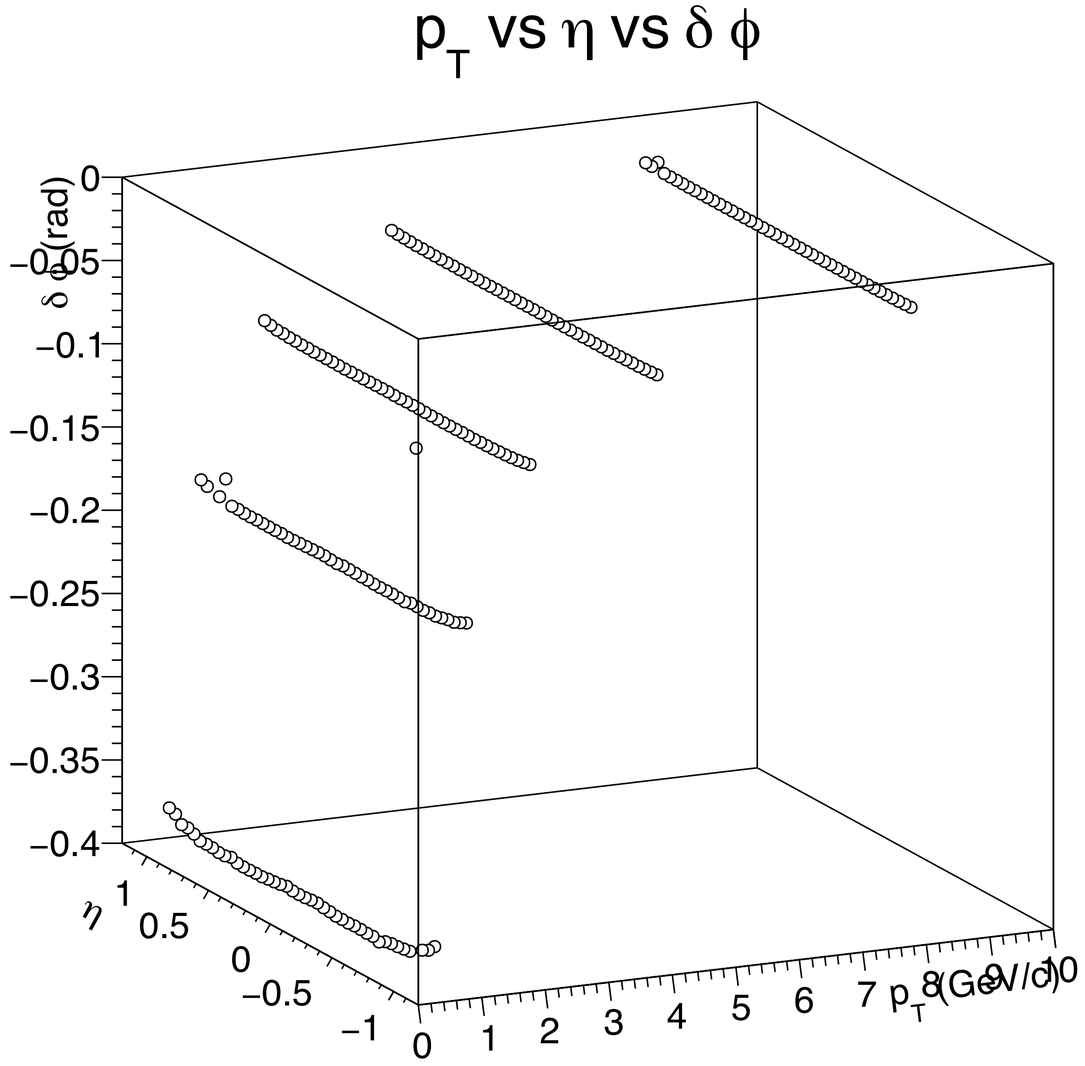
3rd topic to the next topic

```
#pragma once

namespace SiCalo
{
  const double pt_dphi_par0 = -0.1922;

  //! Pt of electron in GeV/c is calculated using given dphi (rad), which is a difference of phi at INTT and ECal, at EMCal surface
  double GetPt( double dphi );

  //! Phi of electron in radian in [0, 2pi] is calculated using givein pT in GeV/c.
  double GetDphi( double pt );

};

#ifdef __CINT__
#include "SiCalo_pt_ideal.cc"
#endif // __CINT__
```

p_T vs $\Delta\phi$



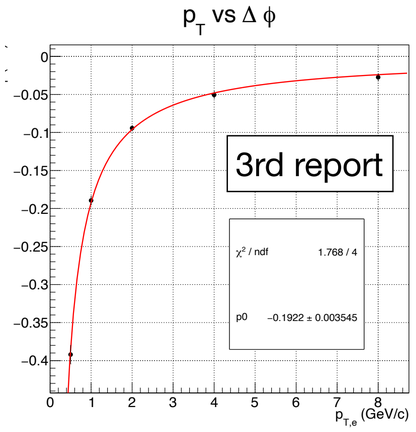
p_T vs η vs $\delta\phi$

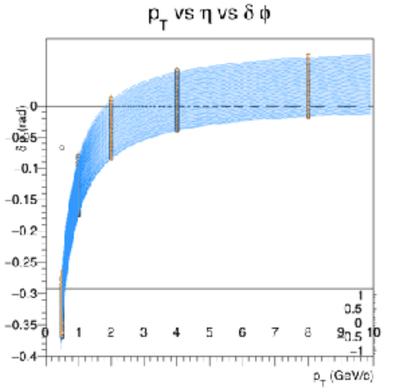


C++ functions to get $p_T/\Delta\phi$ with $\Delta\phi/p_T$ at $\eta = 0$ were provided.

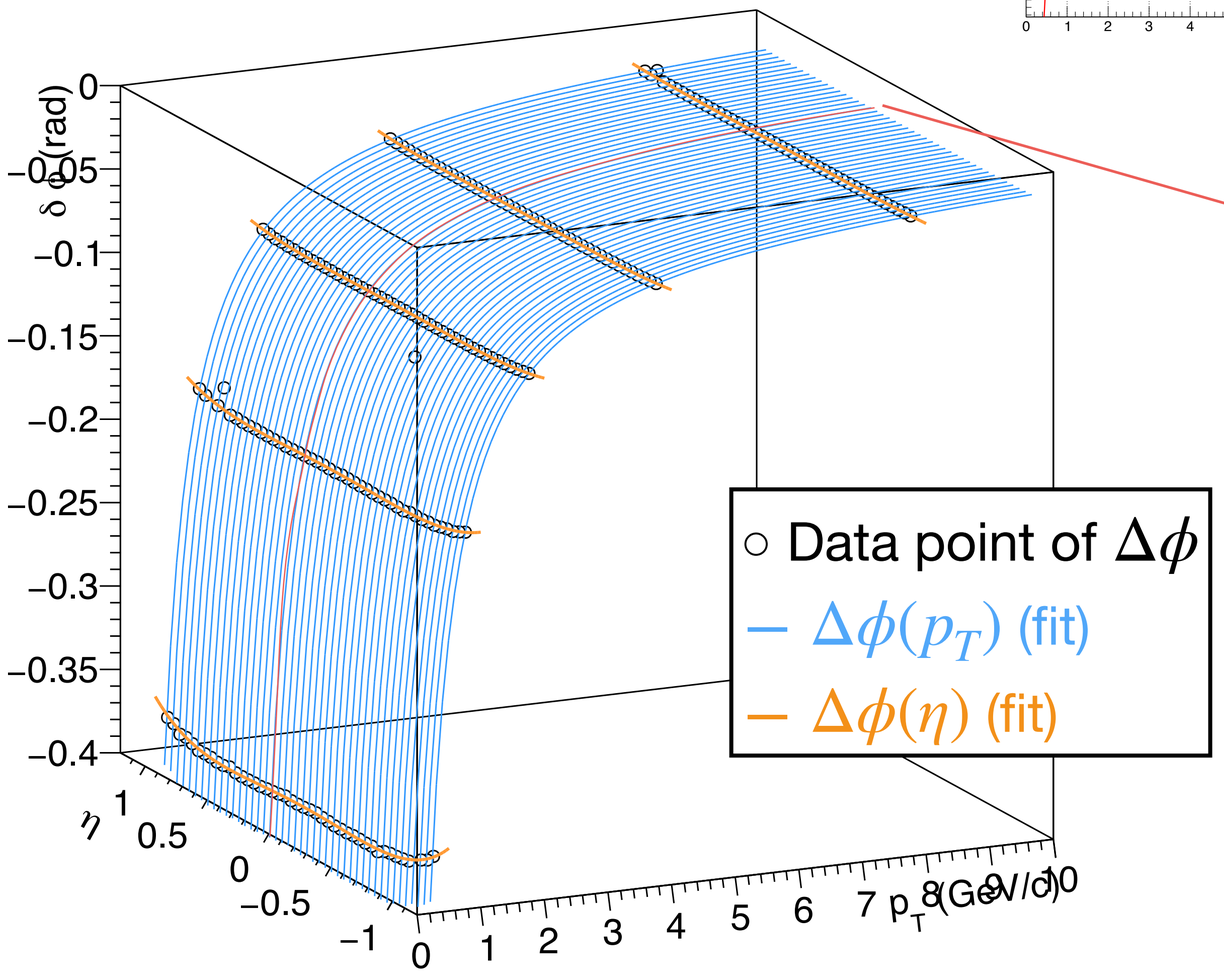
C++ function to provide $\Delta\phi(p_T, \eta, z = 0)$ need to be made.

$\Delta\phi(p_T, \eta)$

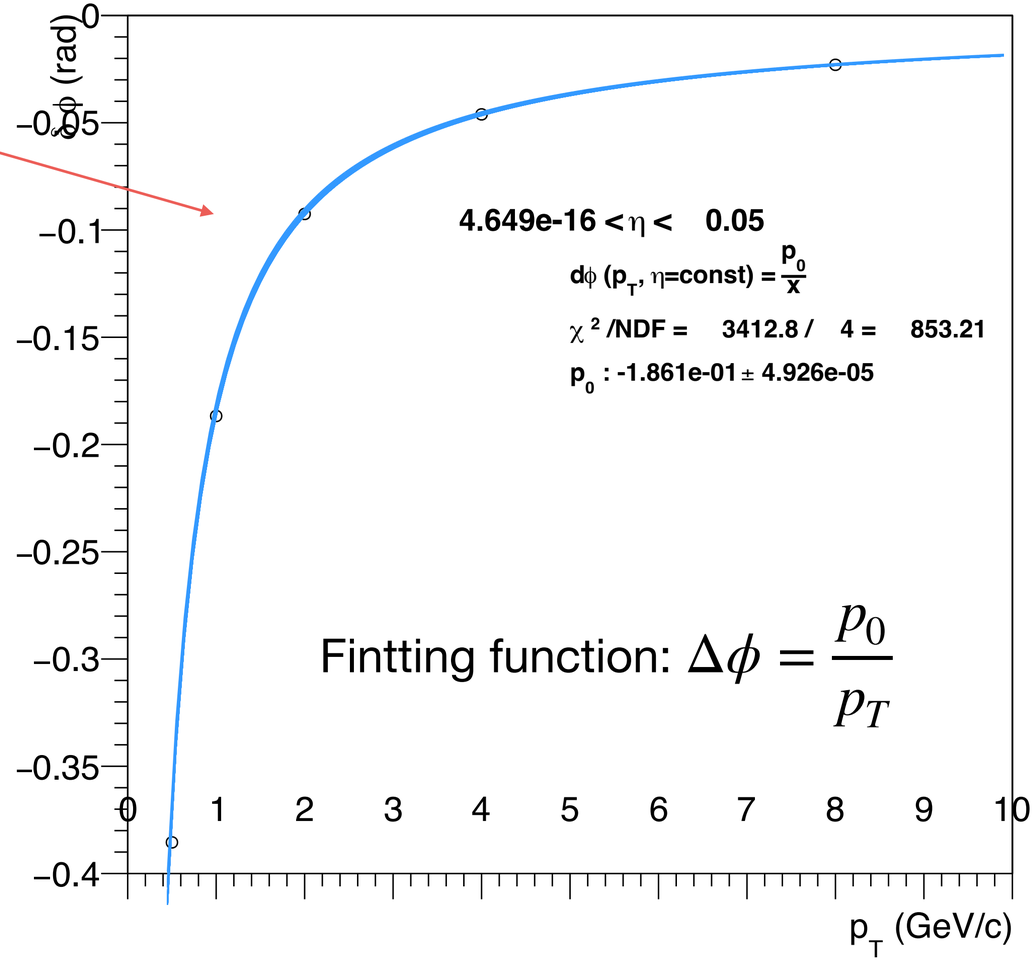




p_T vs η vs δ ϕ

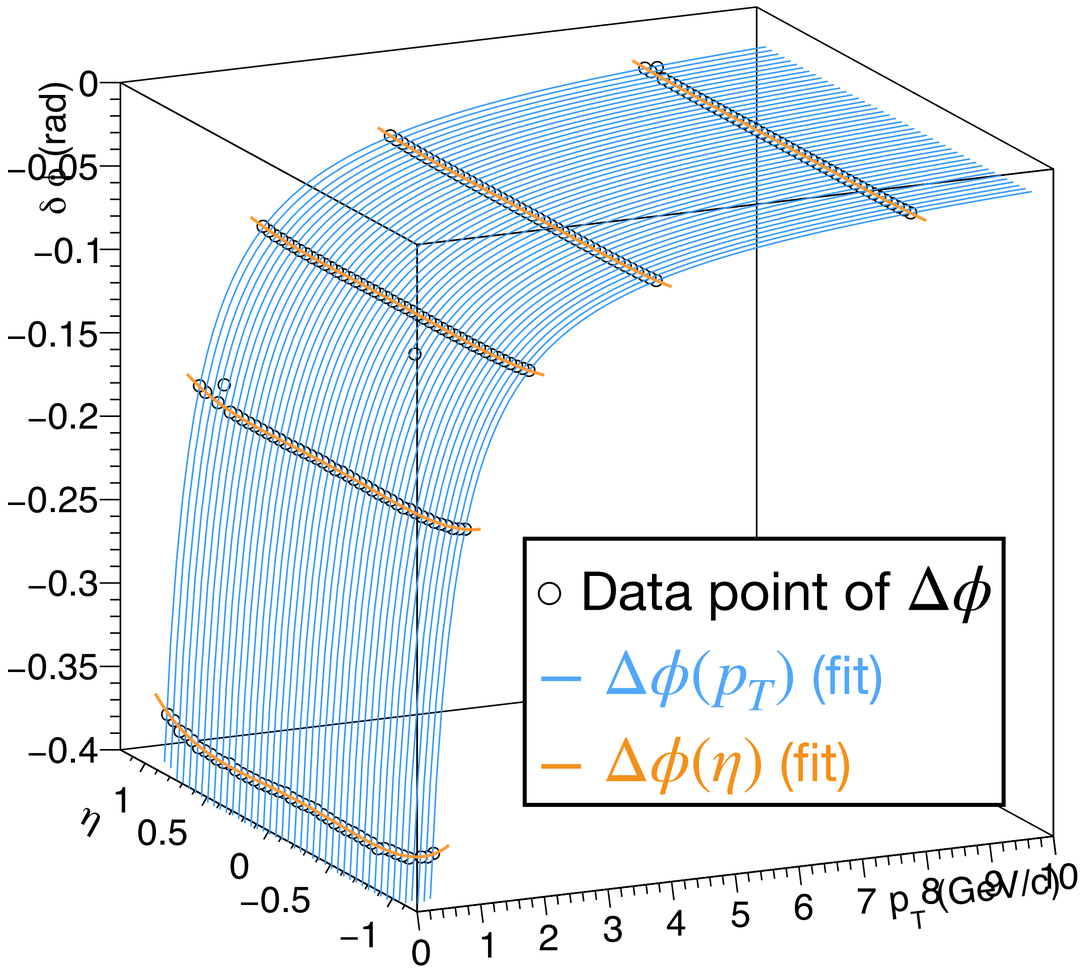


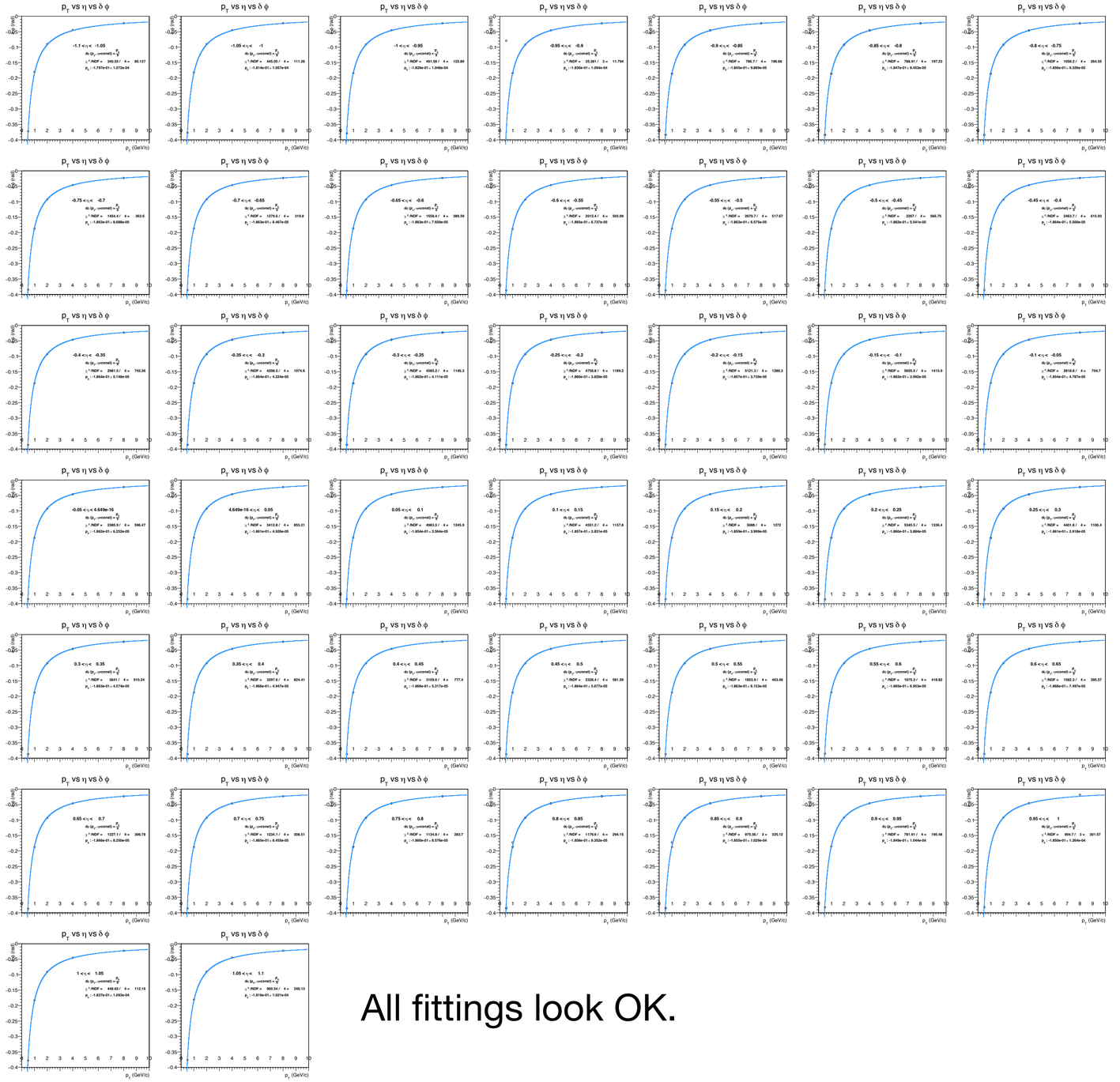
p_T vs η vs δ ϕ



$\Delta\phi(p_T, \eta)$

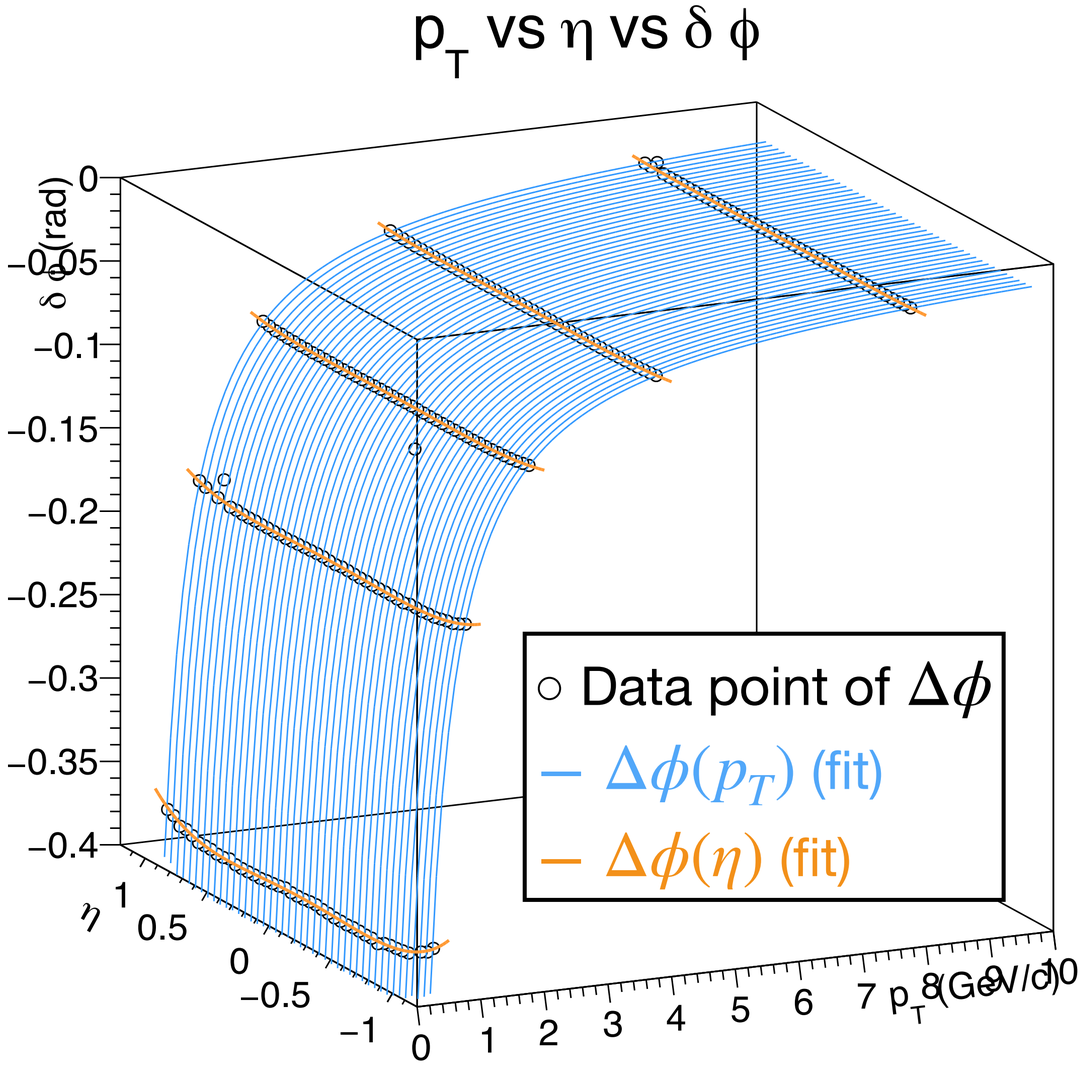
p_T vs η vs $\delta\phi$



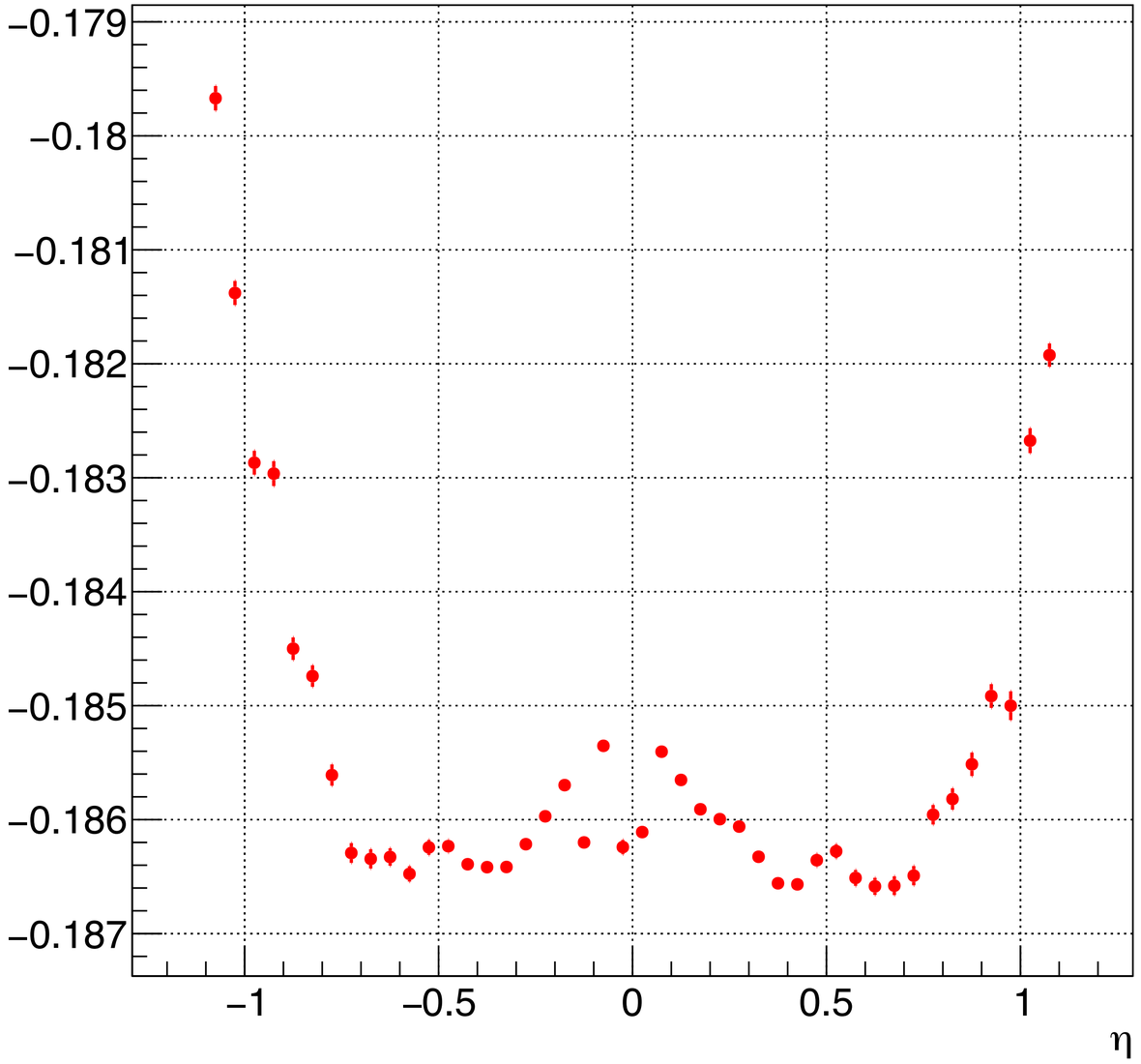


All fittings look OK.

$\Delta\phi(p_T, \eta)$



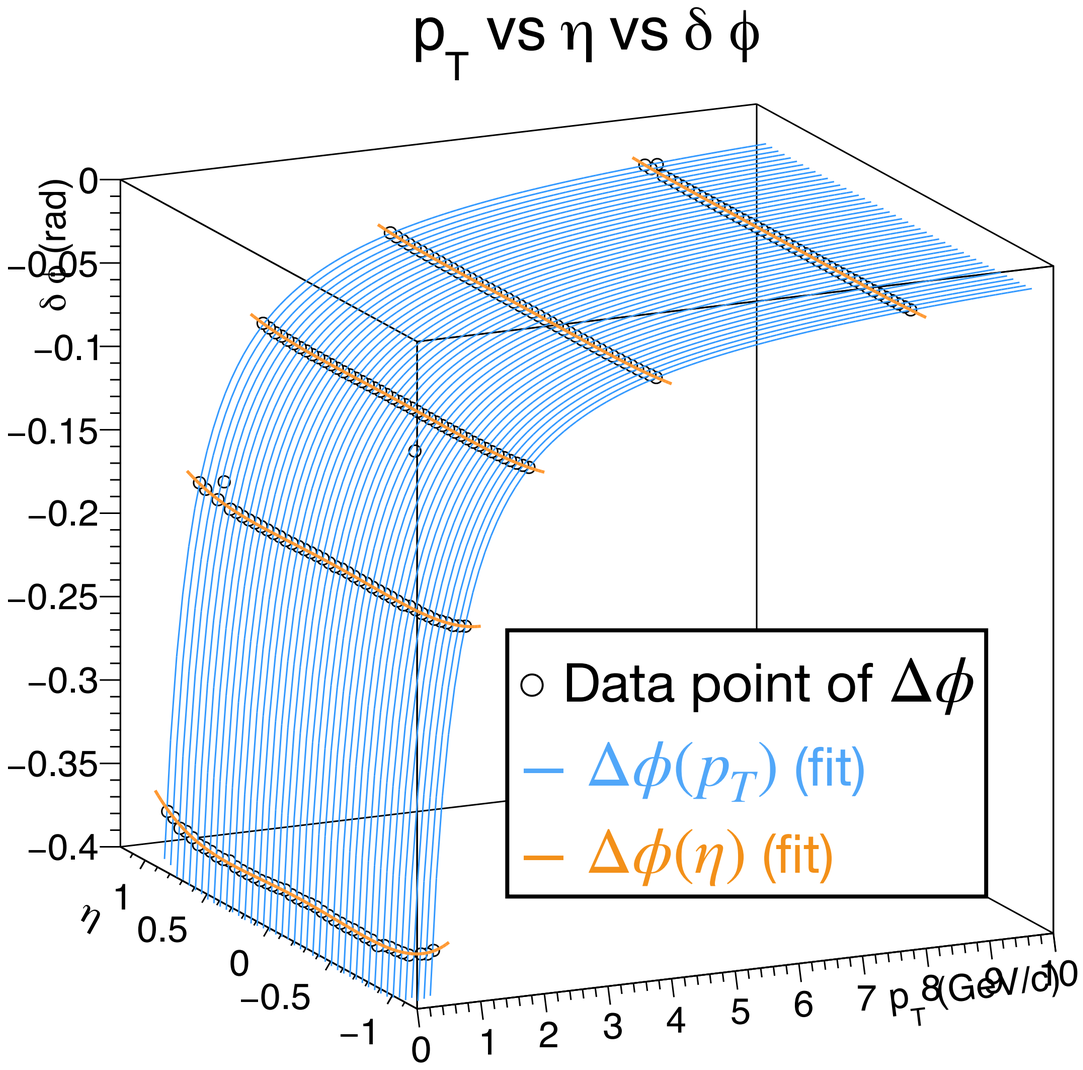
p_0 $d\phi\ (p_T, \eta=\text{const}) = \frac{p_0}{x}$

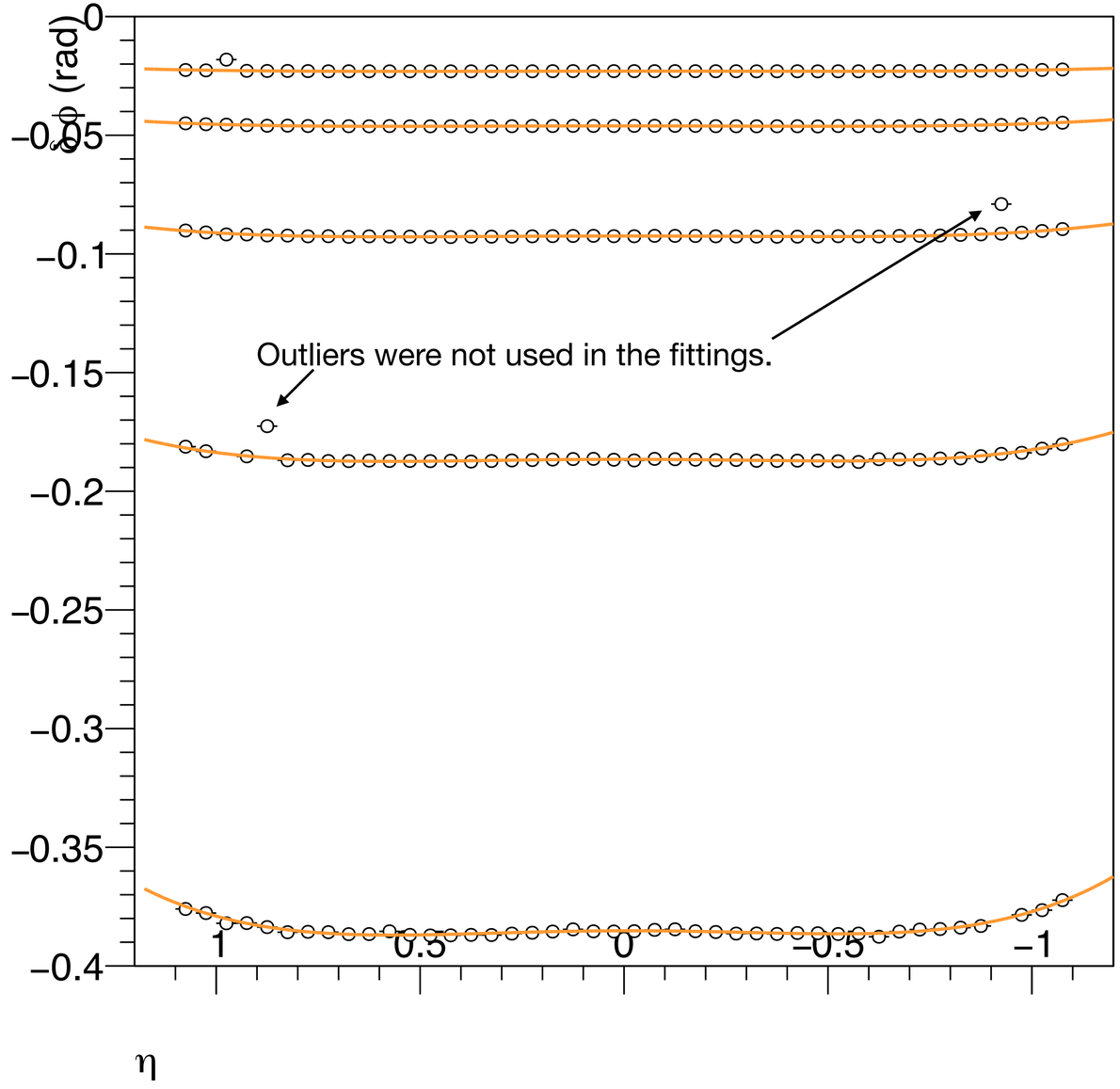


The fitting parameter at each η

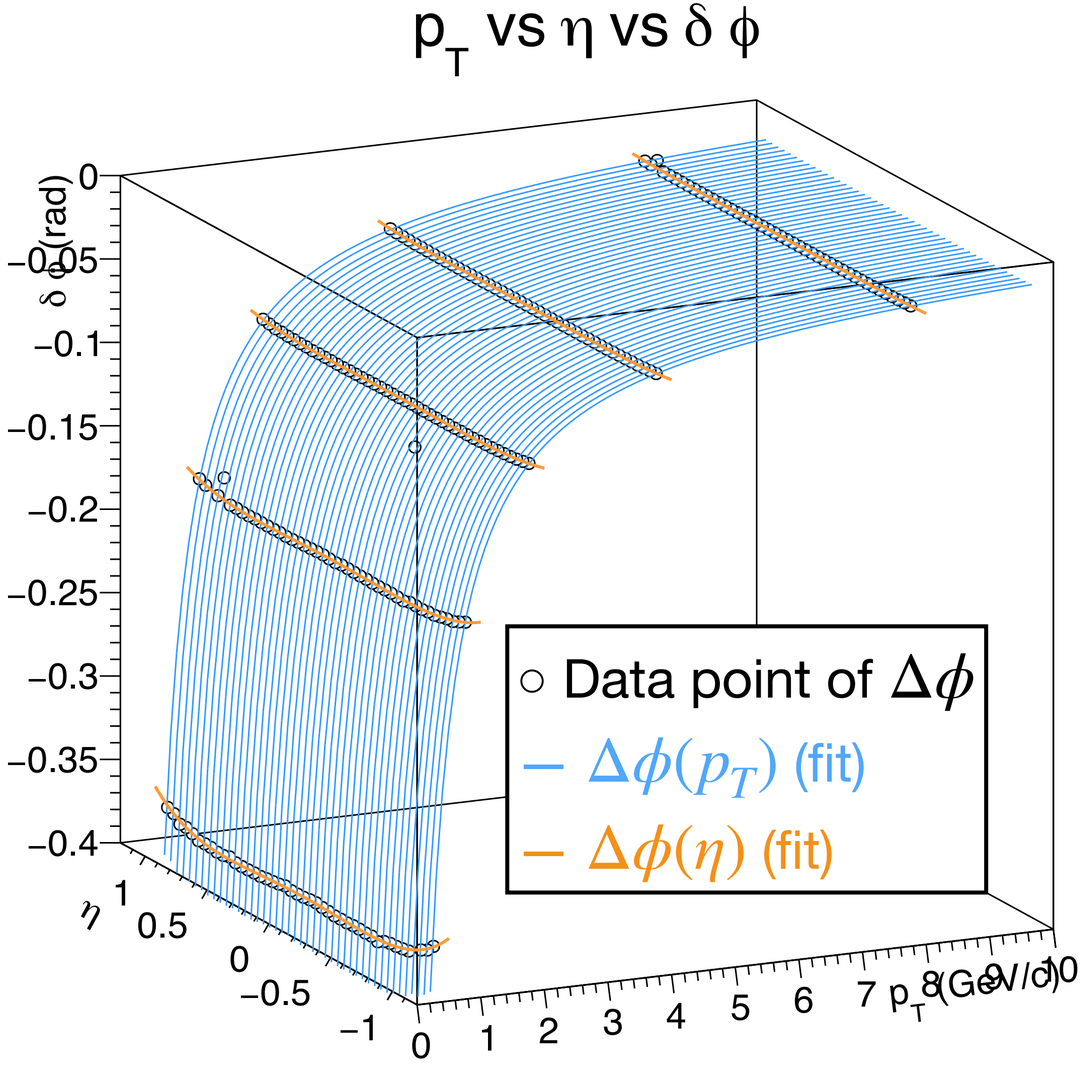
$\Delta\phi(p_T, \eta)$

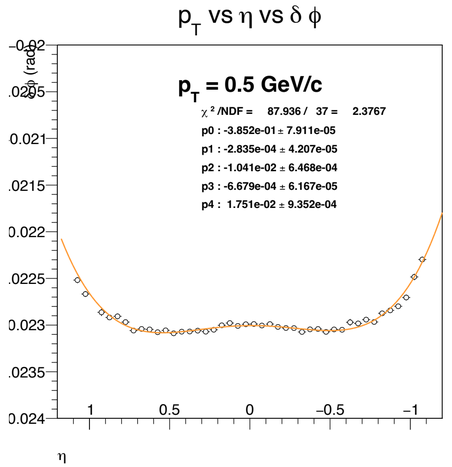
Fintting function: $\Delta\phi = \sum_{i=0}^{4} p_i \eta^i$

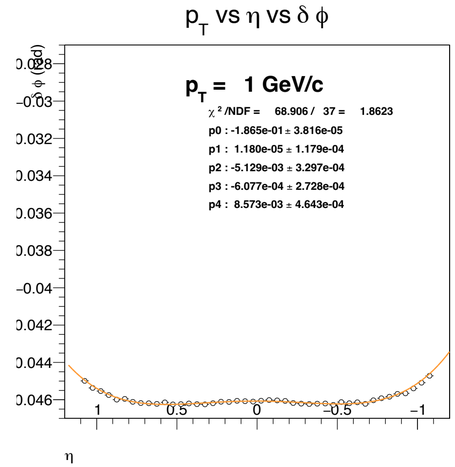


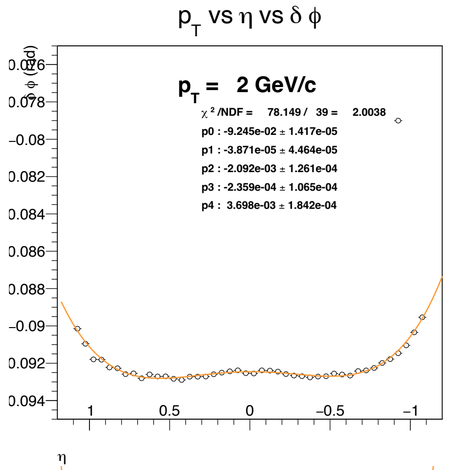


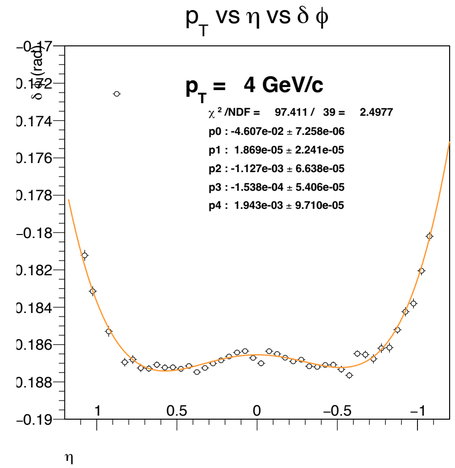
$\Delta\phi(p_T, \eta)$

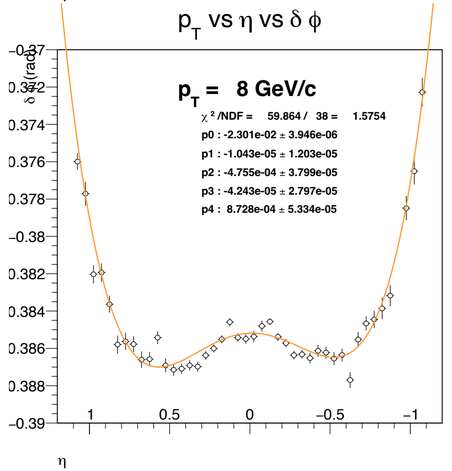






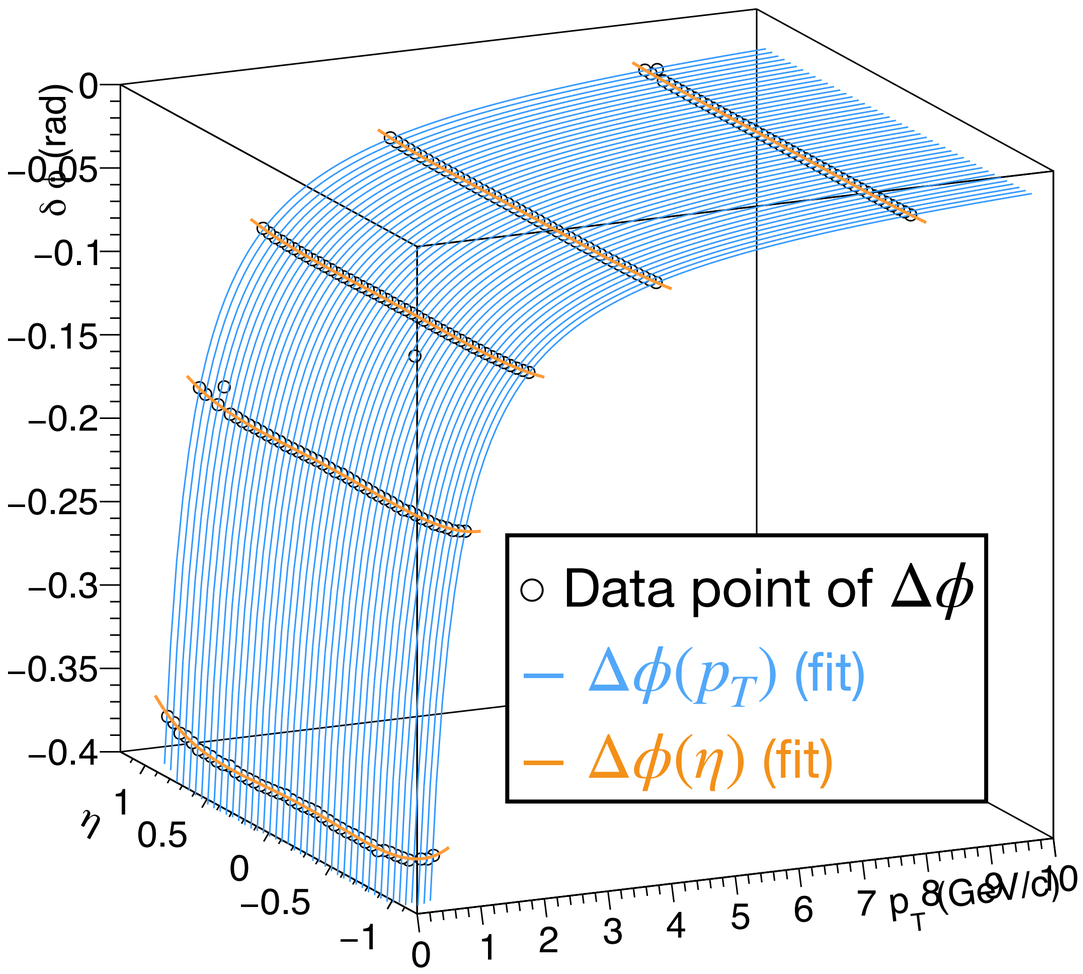


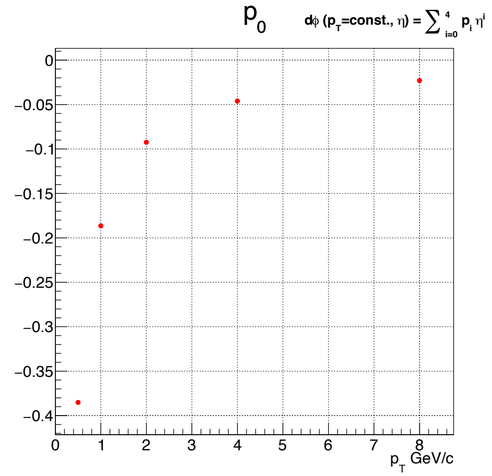


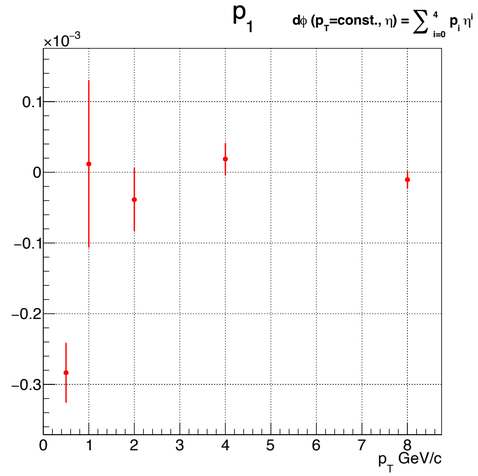


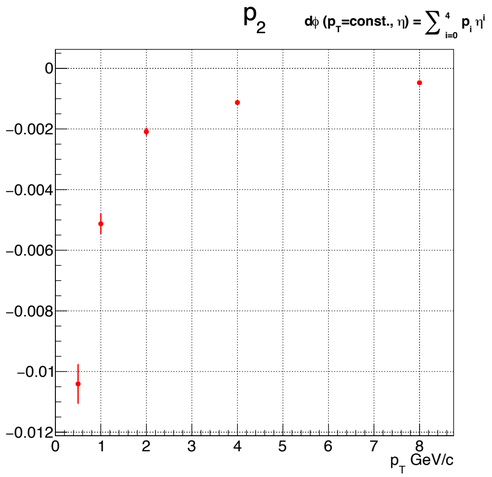
Fintting function: $\Delta\phi = \sum_{i=0}^{4} p_i \eta^i$

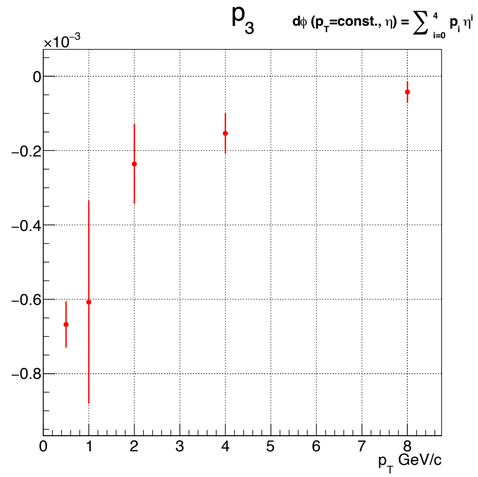
$\Delta\phi(p_T, \eta)$

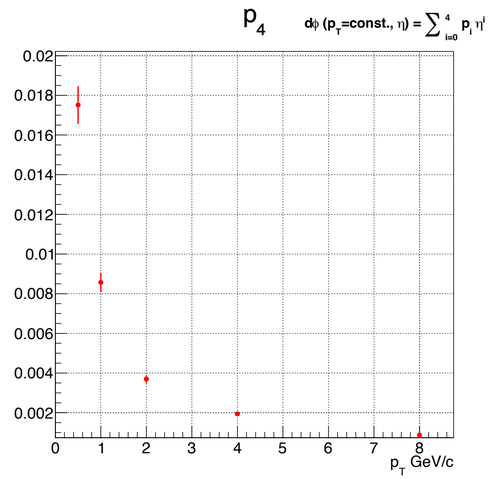








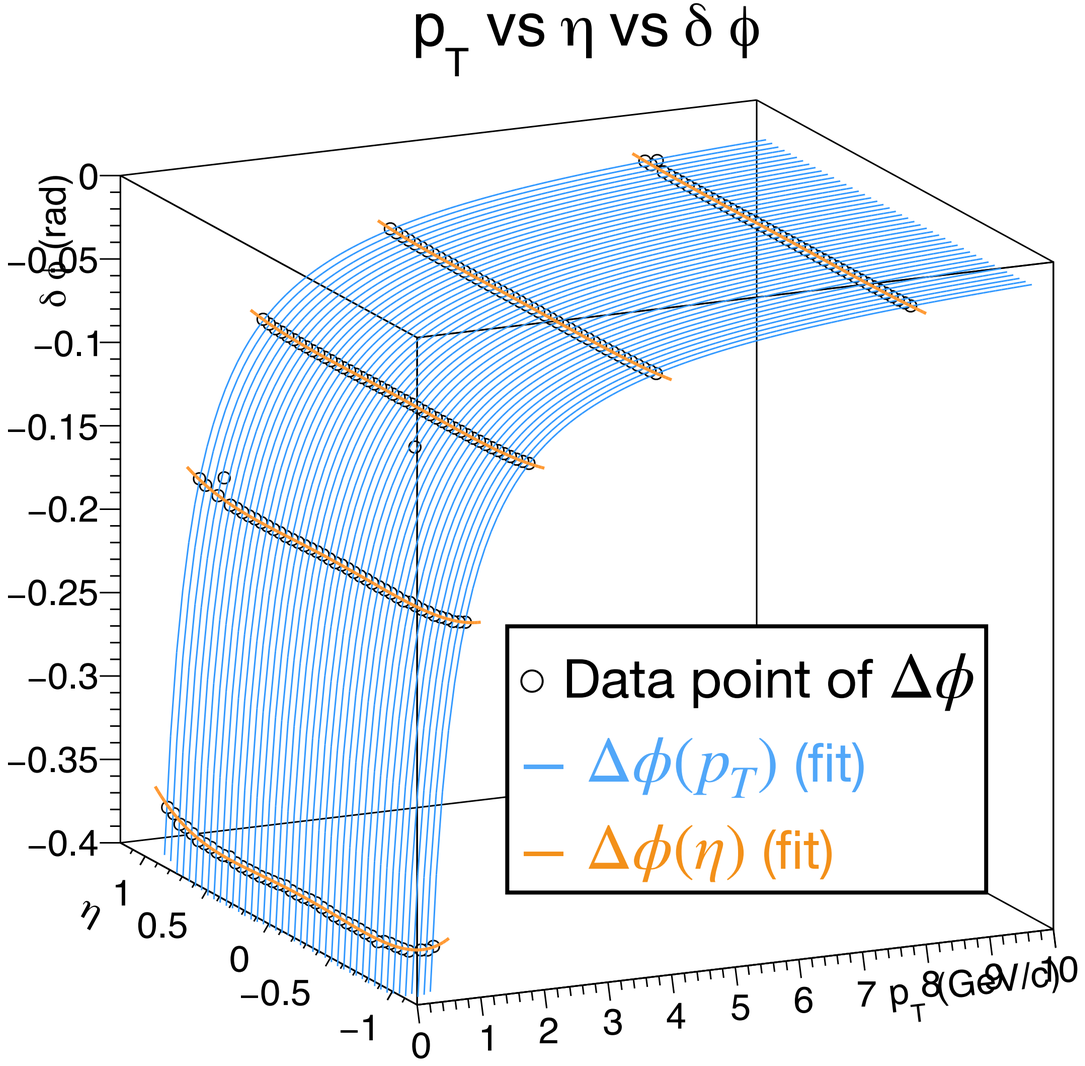


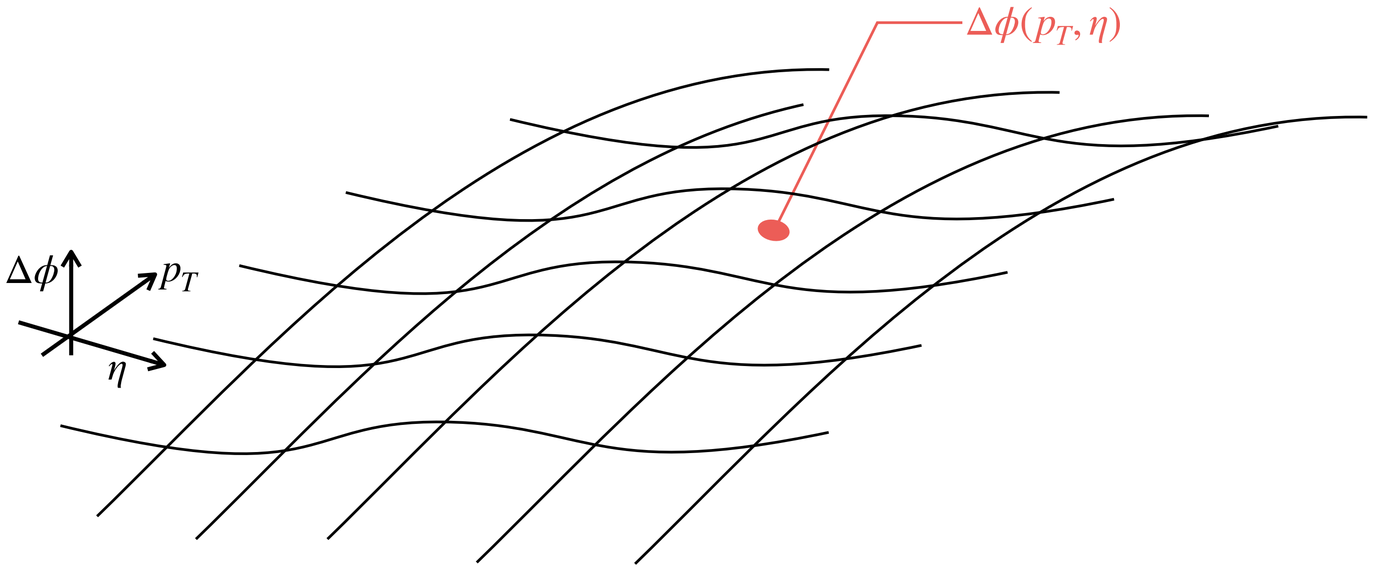


Fitting parameters as a function of p_T.

Fintting function: $\Delta\phi = \sum_{i=0}^{4} p_i \eta^i$

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η



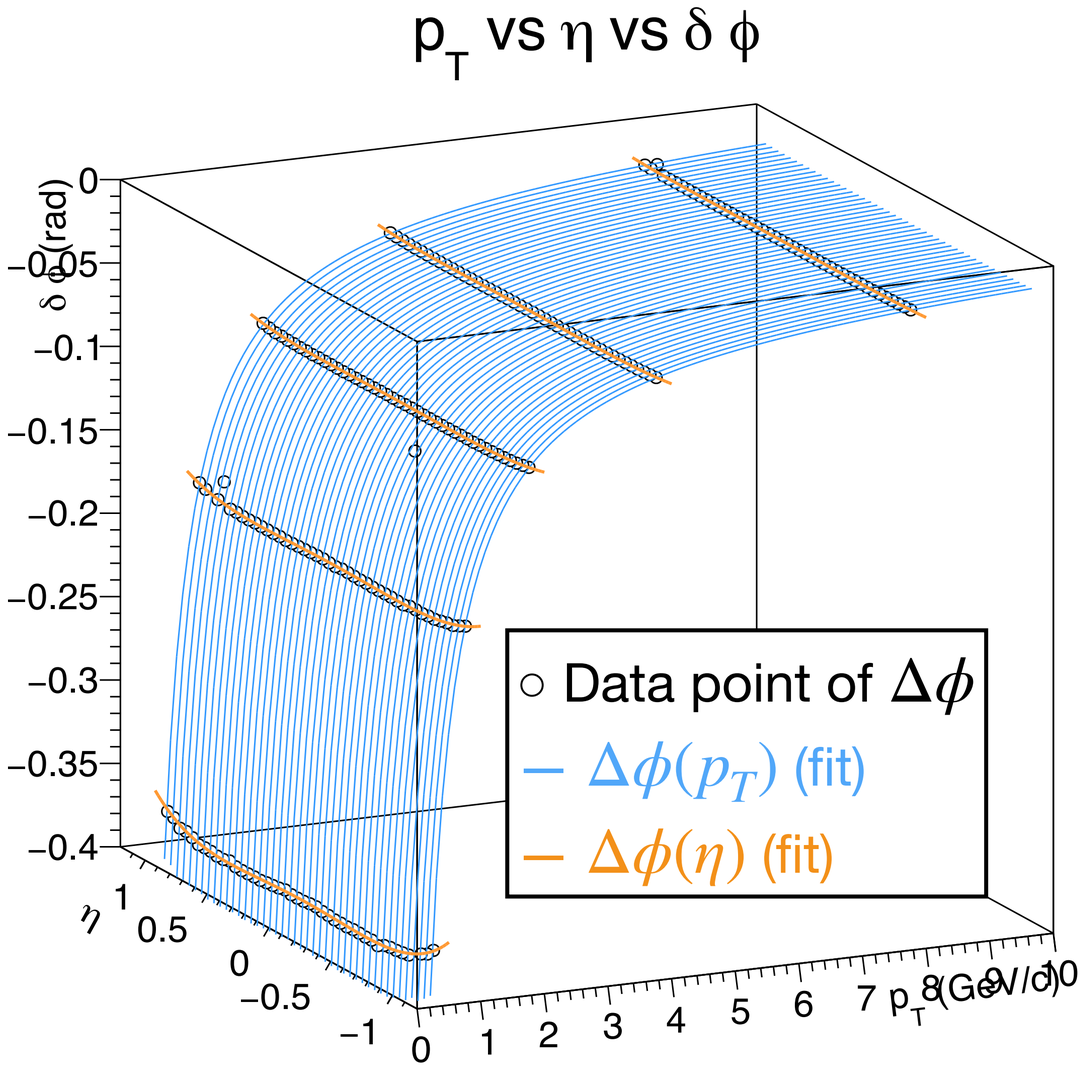


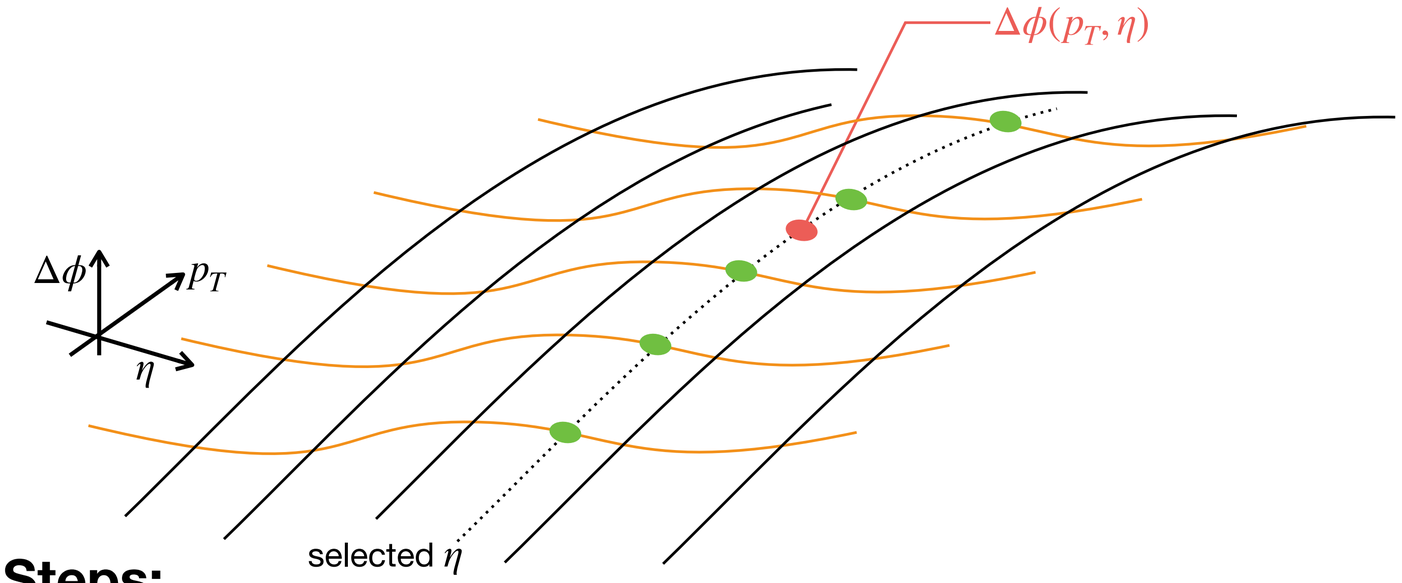
$\Delta\phi$ only on the black lines are known. How can I get $\Delta\phi$ at arbitral p_T and η?

Assumption:

The function form of $\Delta\phi(p_T)$ and $\Delta\phi(\eta)$ are not changed but the parameters depends on η and p_T , respectively.

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η



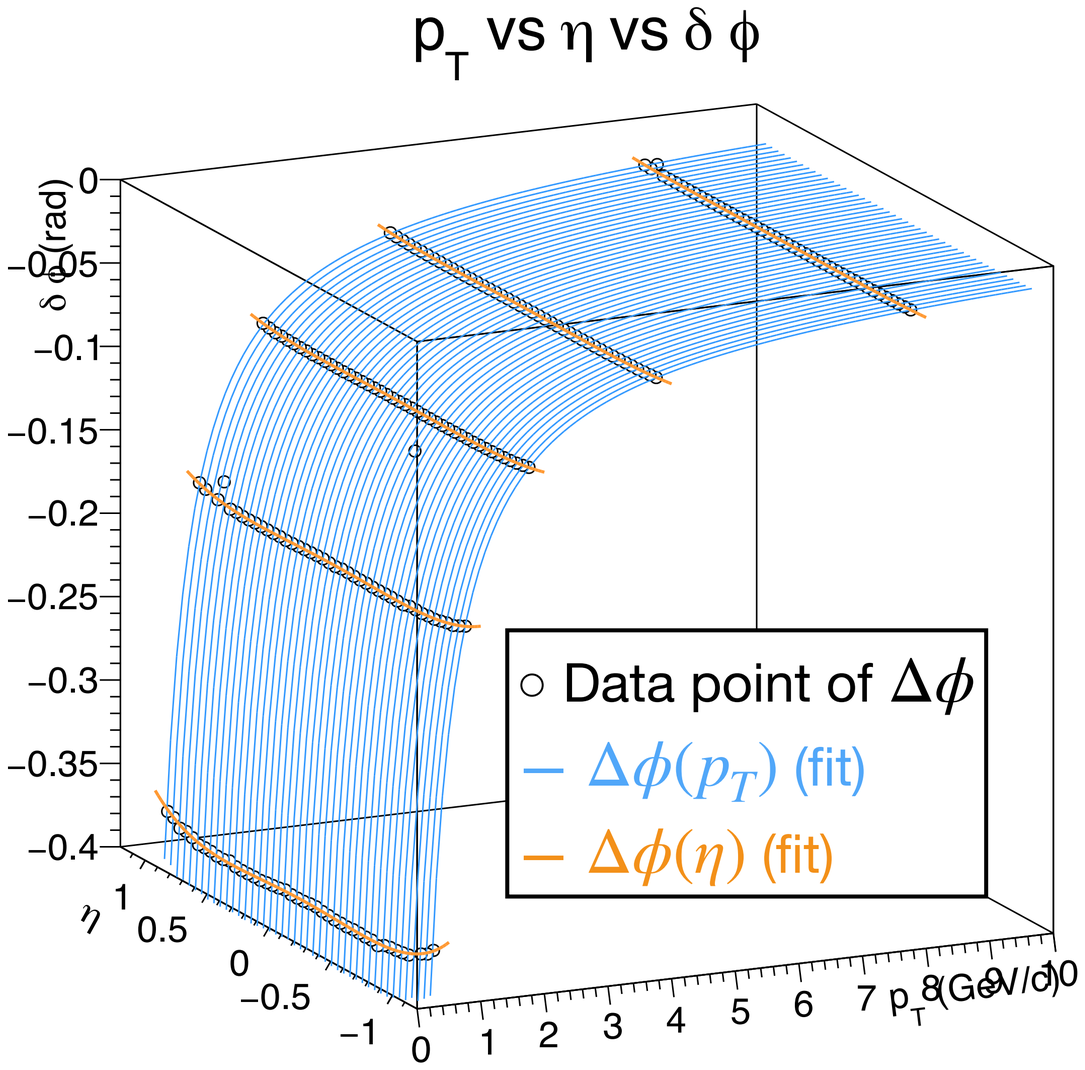


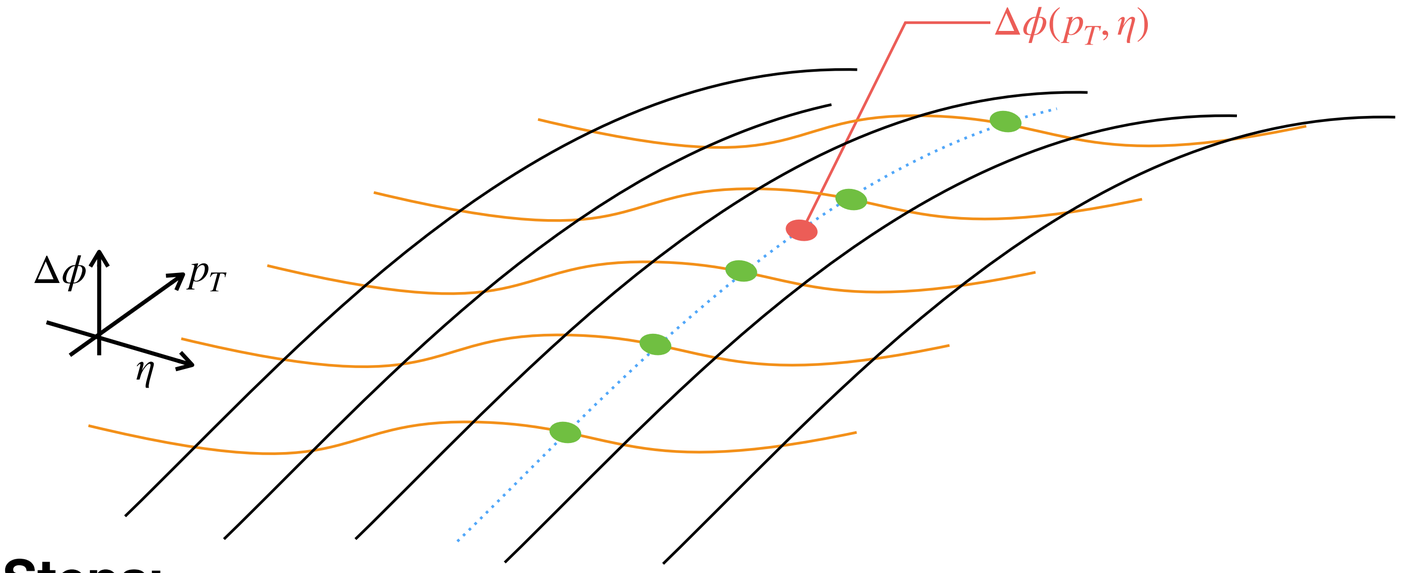
Steps:

1. Get $\Delta\phi$ at the selected η using the fitting functions

$$\Delta\phi(p_T = \text{const}, \eta) = \sum_{i=0}^{4} p_{\eta,i}\ \eta^i$$

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η





Steps:

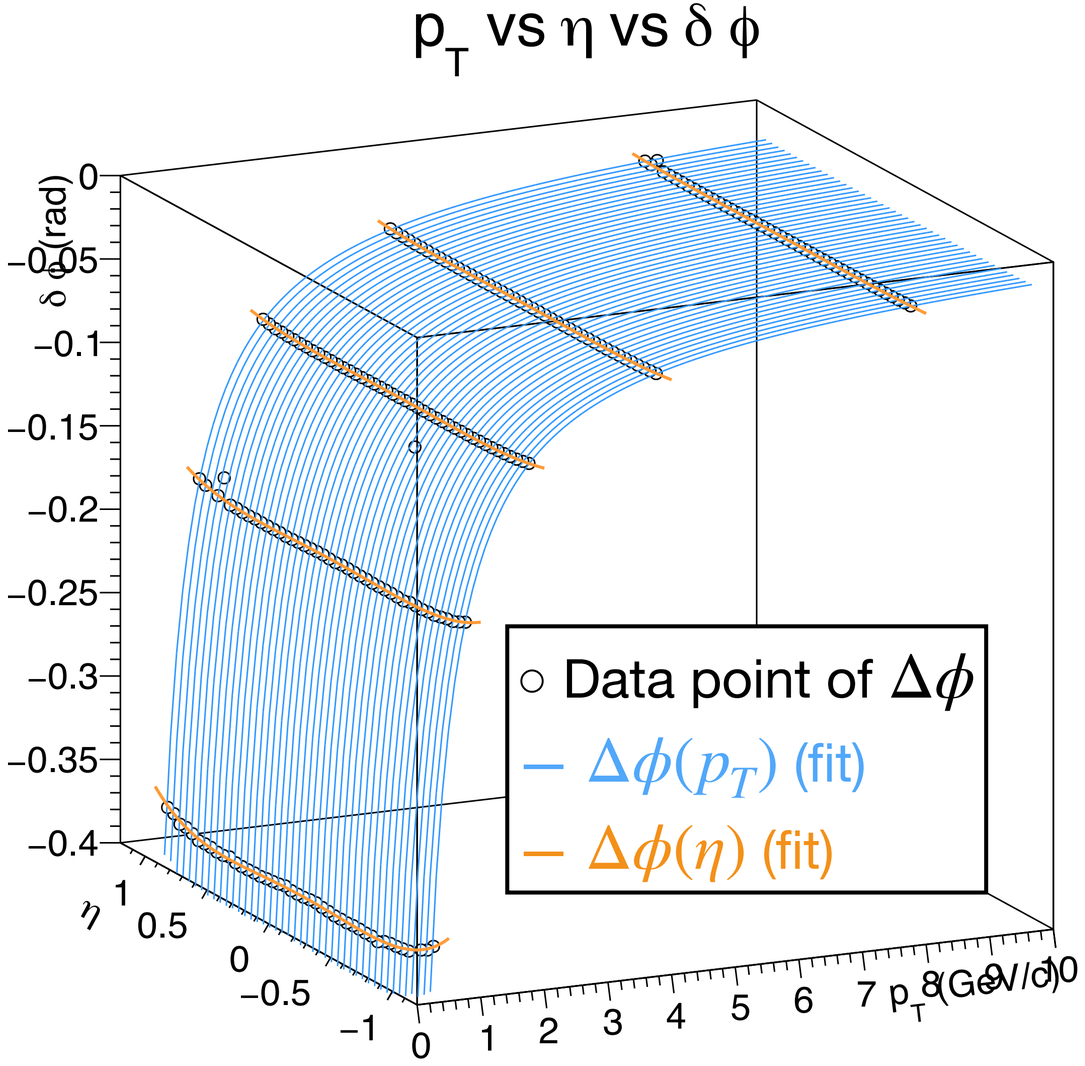
1. Get $\Delta\phi$ at the selected η using the fitting functions

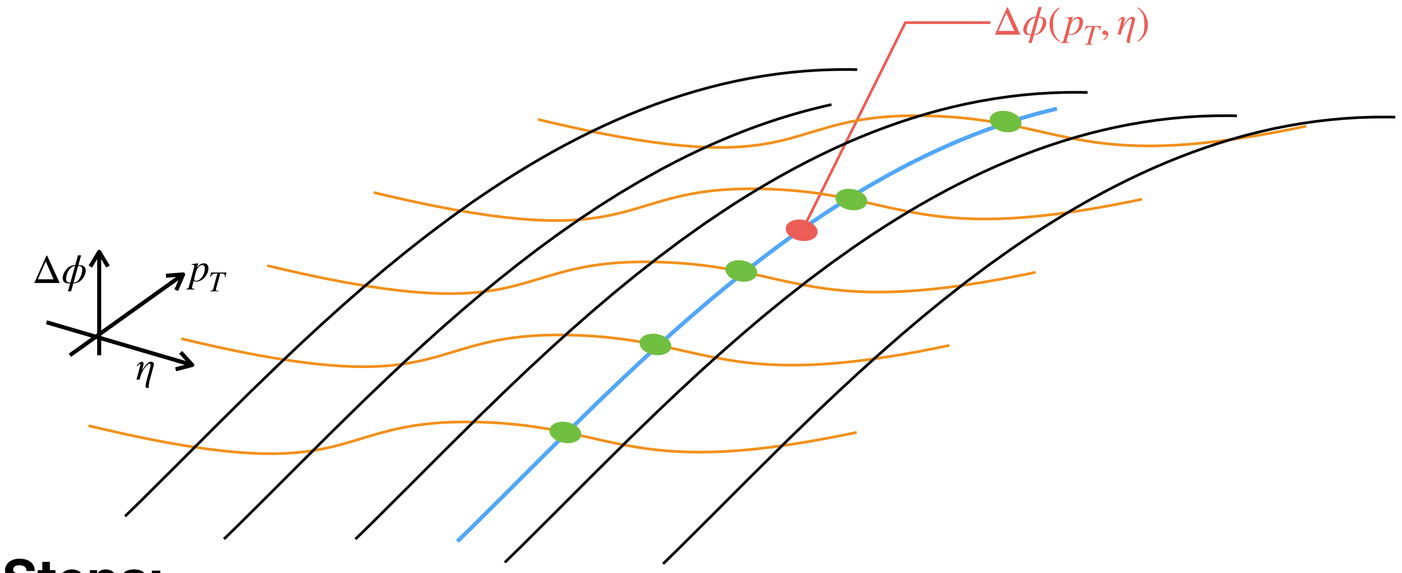
$$\Delta\phi(p_T = \text{const}, \eta) = \sum_{i=0}^{4} p_{\eta,i}\ \eta^i$$

2. Make the graph of $\Delta\phi$ and fit with

$$\Delta\phi(p_T, \eta = \text{const.}) = \frac{p_{p_T}}{p_T}$$

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η





Steps:

1. Get $\Delta\phi$ at the selected η using the fitting functions

$$\Delta\phi(p_T = \text{const}, \eta) = \sum_{i=0}^{4} p_{\eta,i}\ \eta^i$$

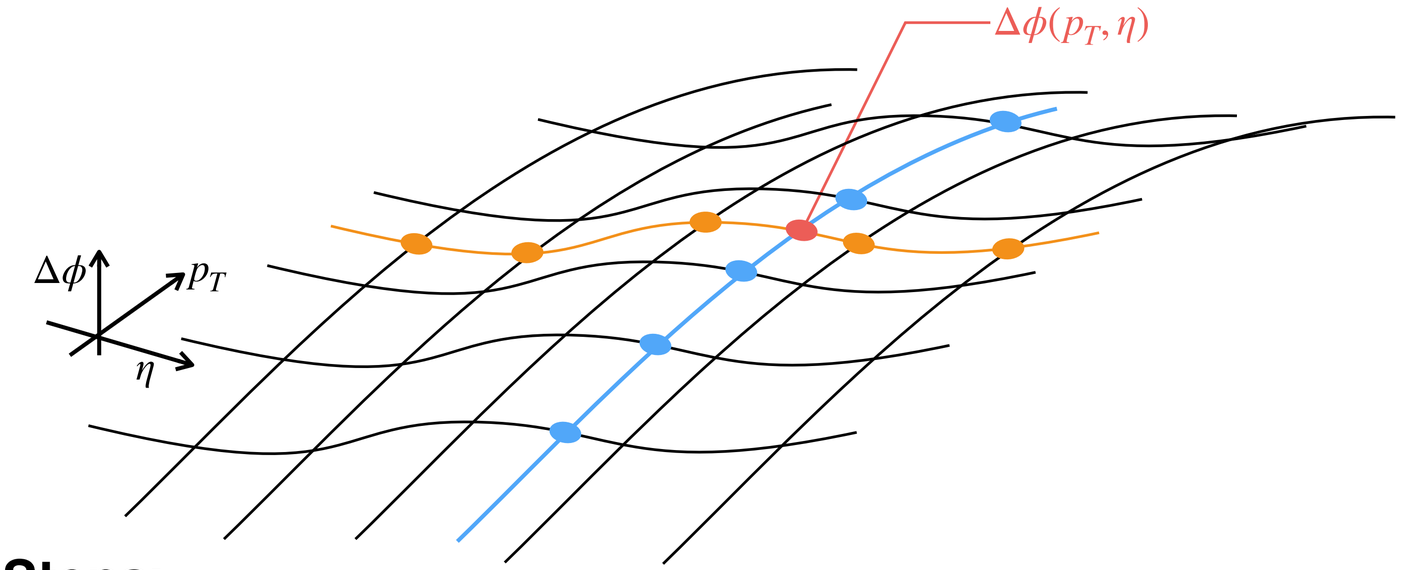
2. Make the graph of $\Delta\phi$ and fit with

$$\Delta\phi(p_T, \eta = \text{const.}) = \frac{p_{p_T}}{p_T}$$

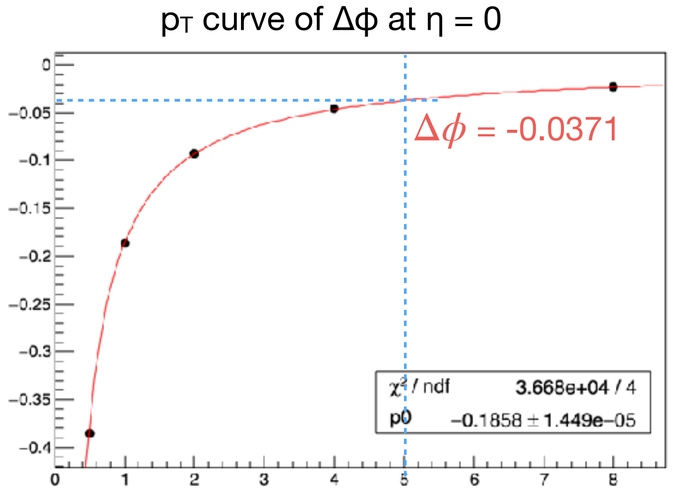
3. Get $\Delta\phi$ using the fitting function $\Delta\phi(p_T, \eta = \text{const.}) = \frac{p_{p_T}}{p_T}$

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η

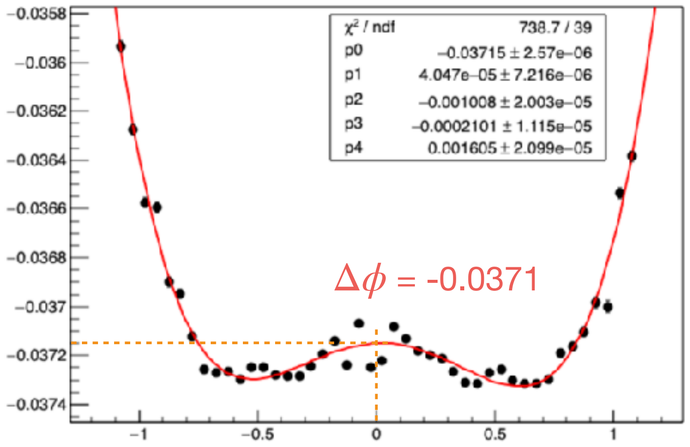
Example: $\Delta\phi(p_T = 5\ \text{GeV}, \eta = 0)$



Method1:
First, η interpolation,
then p_T interpolation



Method2:
First, p_T interpolation,
then η interpolation



Steps:

1. Get $\Delta\phi$ at the selected η using the fitting functions

$$\Delta\phi(p_T = \text{const}, \eta) = \sum_{i=0}^{4} p_{\eta,i}\ \eta^i$$

2. Make the graph of $\Delta\phi$ and fit with

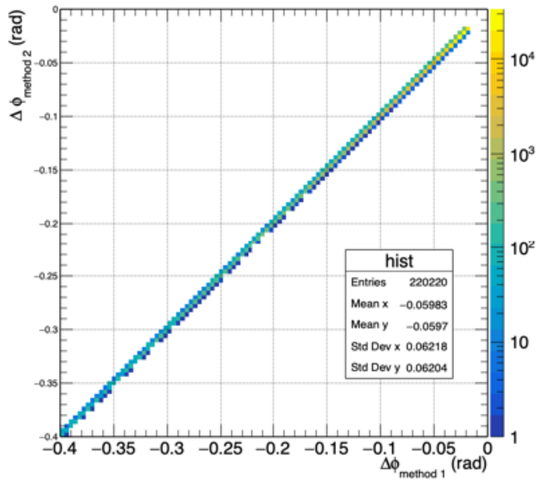
$$\Delta\phi(p_T, \eta = \text{const.}) = \frac{p_{p_T}}{p_T}$$

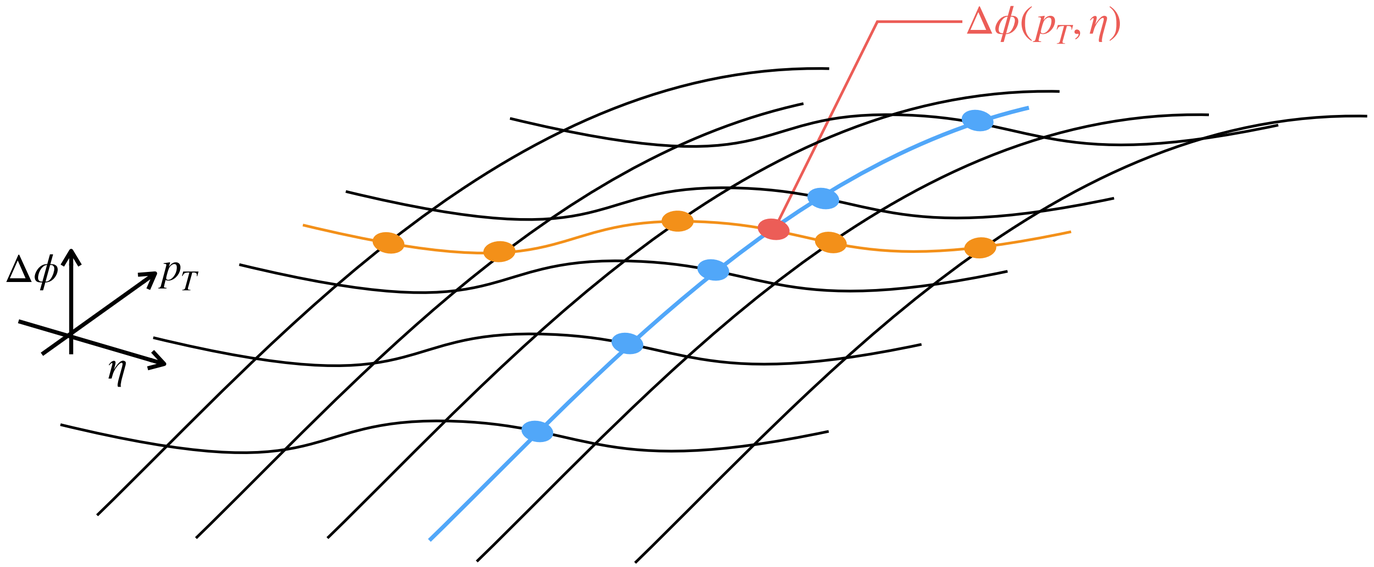
3. Get $\Delta\phi$ using the fitting function $\Delta\phi(p_T, \eta = \text{const.}) = \frac{p_{p_T}}{p_T}$

Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η

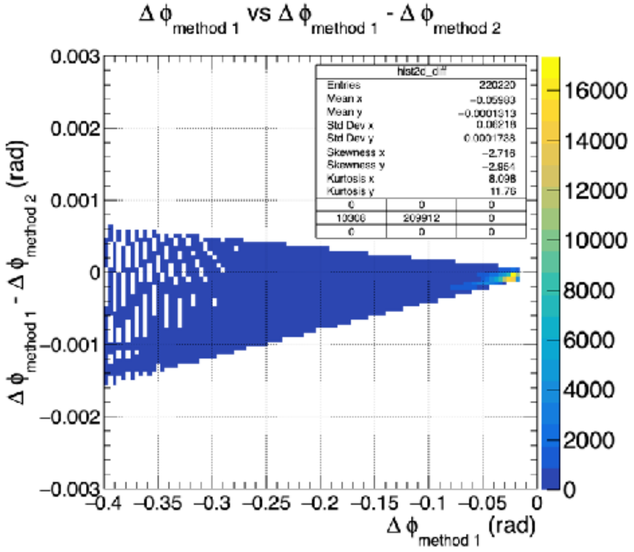
Comparison of the methods

Method1 vs Method2

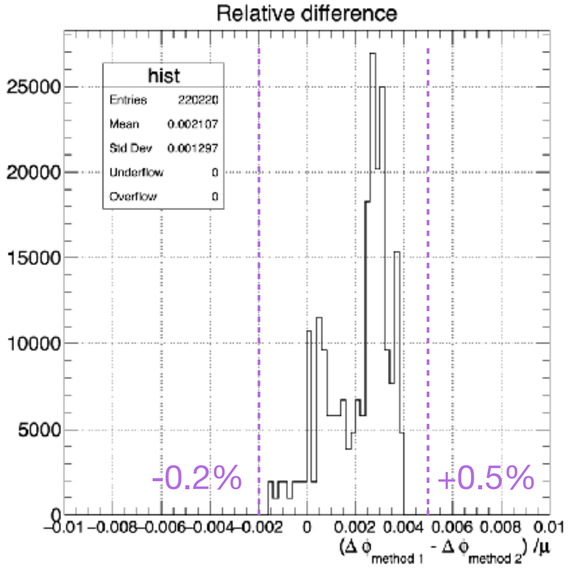




Method1 vs Δ



Δ / average

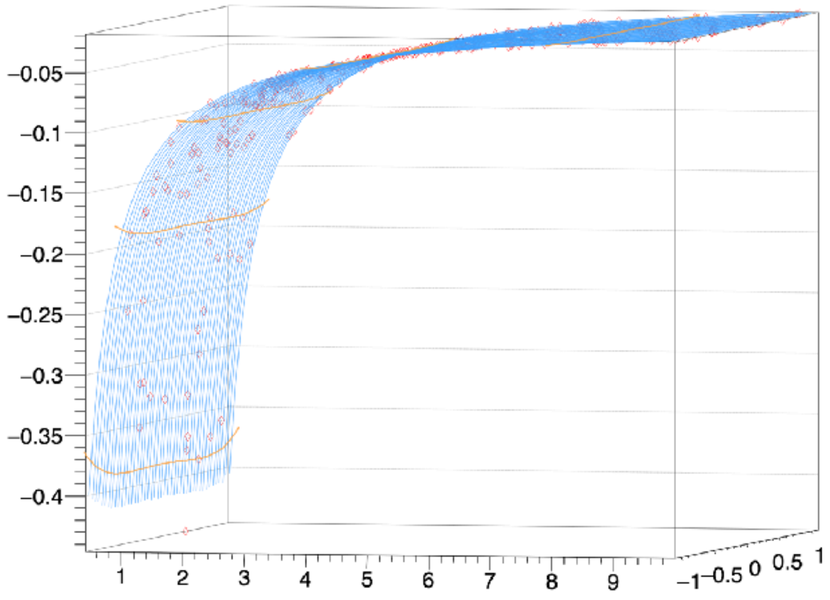


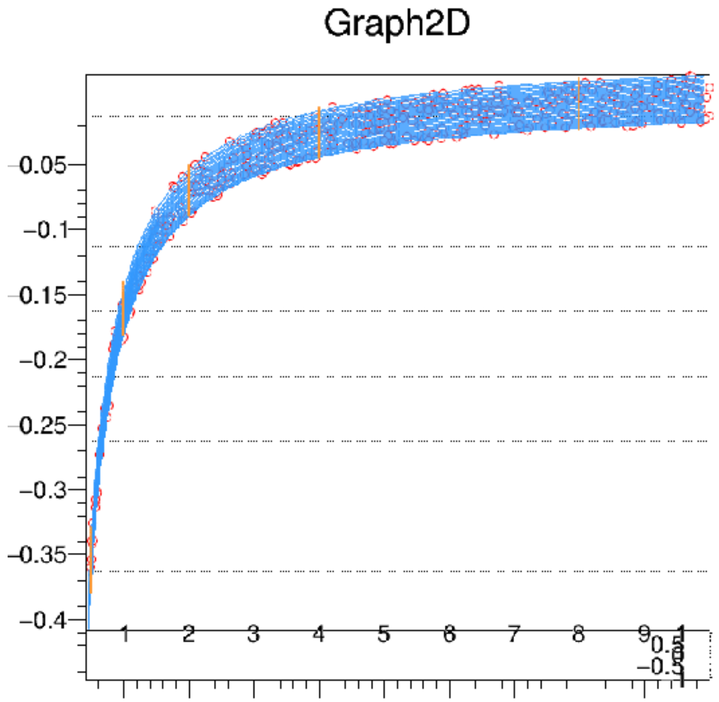
Comparison b/w the 2 methods was performed with sample data points:

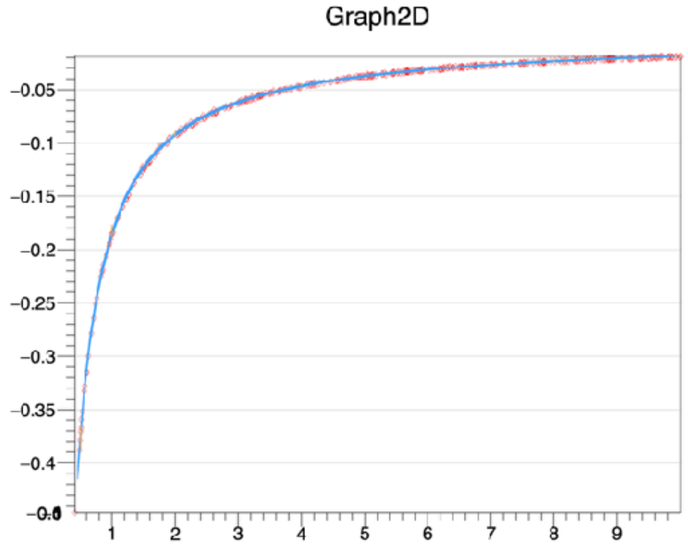
- p_T = [0.4, 10] GeV/c, 0.01 GeV/c step
- η = [-1.1, 1.1], 0.01 step

A good agreement b/w 2 methods, within 0.5%, was confirmed. Though this agreement doesn't guarantee the correctness of the interpolation method, I think it's good enough to use.

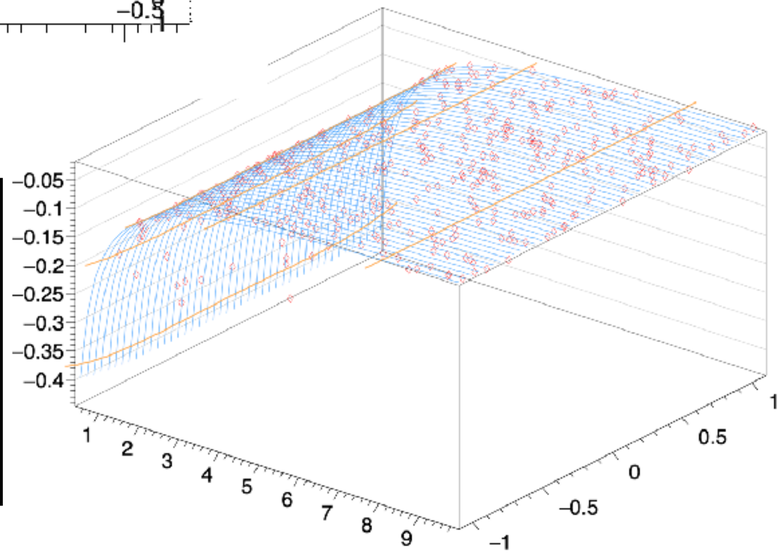
Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η







- $\Delta\phi(p_T)$ (fit)
- $\Delta\phi(\eta)$ (fit)
- Interpolated points (average of the 2 methods)

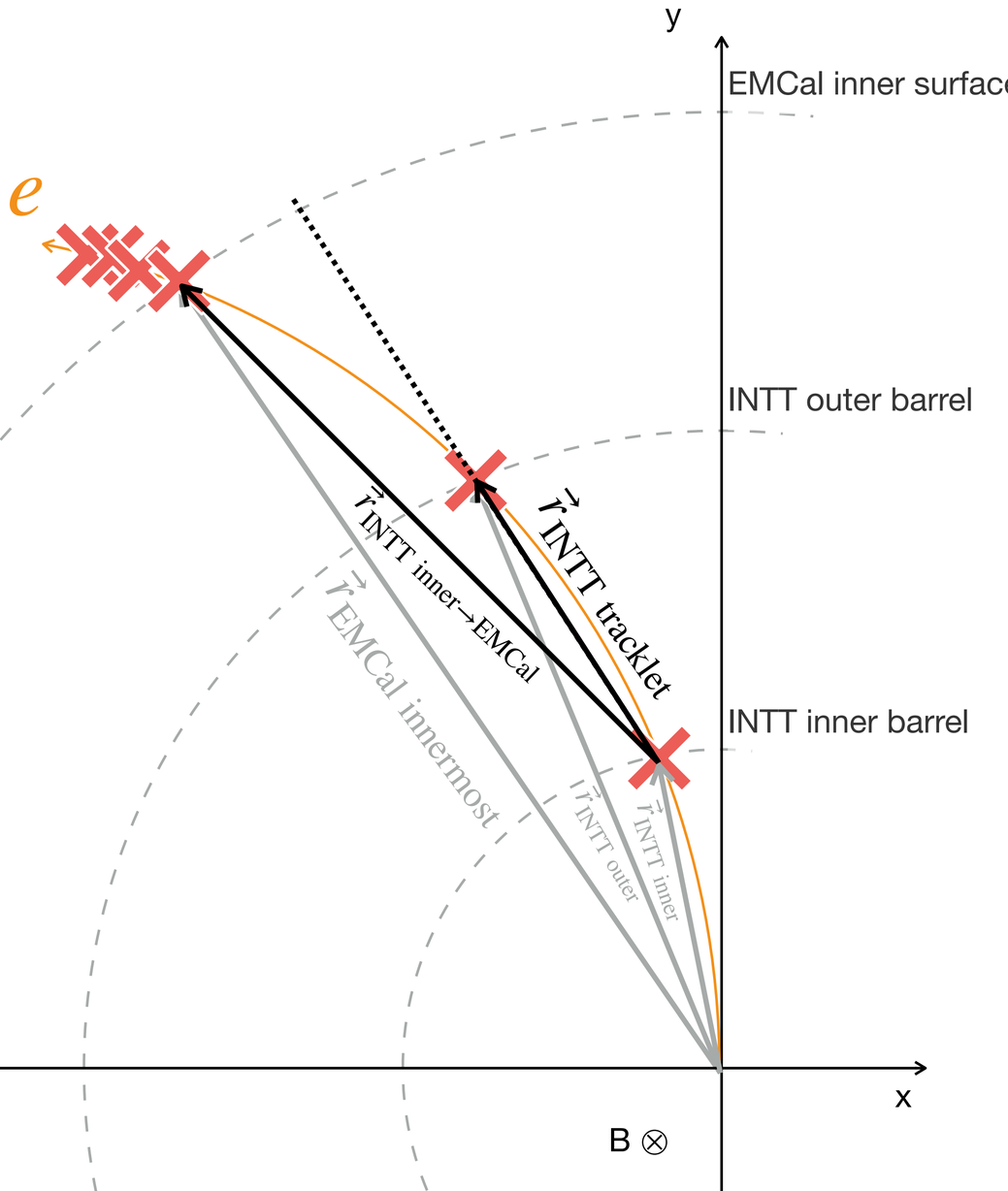


Implementation to provide $\Delta\phi(p_T, \eta)$ with arbitral p_T and η

not done yet…

backup

Basic idea of angular resolution evaluation



$$\vec{r} = \vec{r}_{\text{mes.}} - \vec{r}_{\text{vertex}}\Big|_{\vec{r}_{\text{vertex}}=(0,0,0)} = \vec{r}_{\text{mes.}}$$

EMCal hit: the innermost (smallest ρ) G4Hit on EMCal

$$\vec{r}_{\text{EMCal innermost}}$$

INTT tracklet vector: a line from G4Hit on the inner barrel to that on the outer

$$\vec{r}_{\text{INTT tracklet}} = \vec{r}_{\text{INTT outer}} - \vec{r}_{\text{INTT inner}}$$

A vector from INTT inner barrel to EMcal hit:

$$\vec{r}_{\text{inner}\to\text{EMCal}} = \vec{r}_{\text{EMCal}} - \vec{r}_{\text{inner}}$$

Angle resolution evaluator:

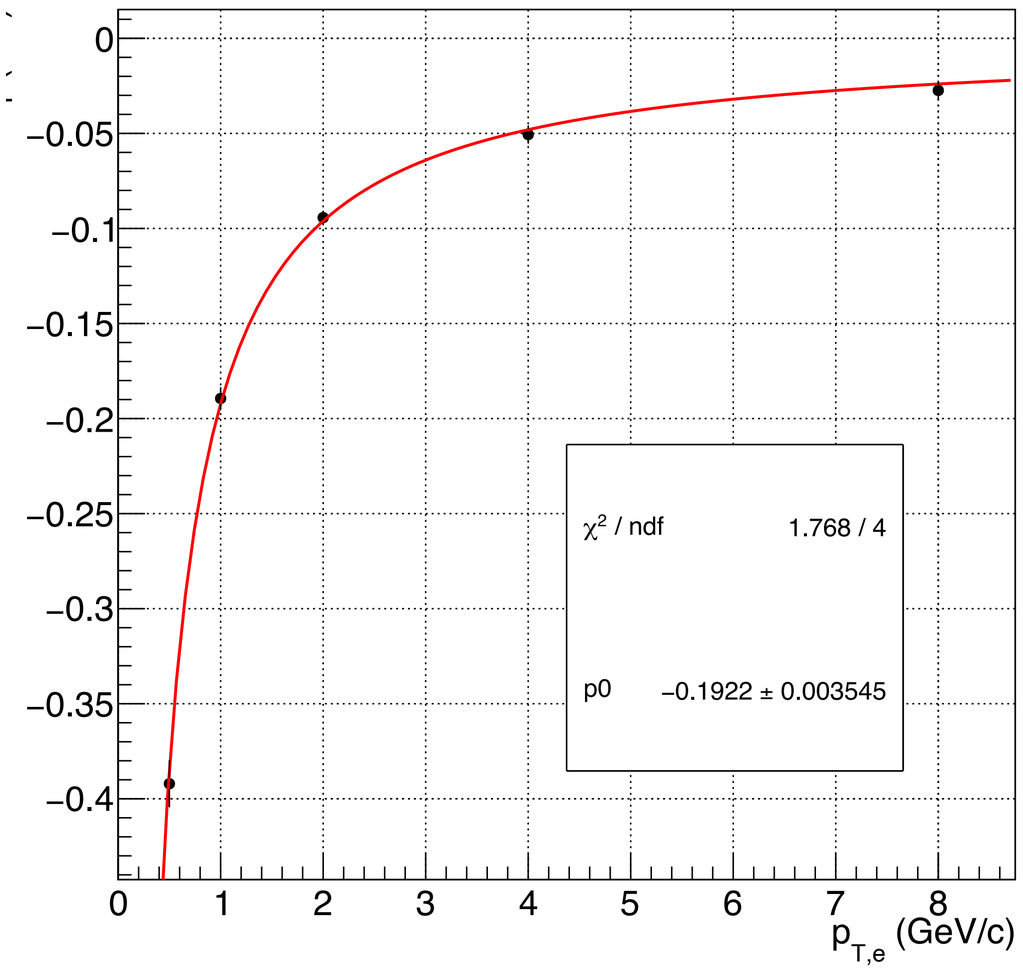
$$\Delta\phi = \phi_{\text{inner}\to\text{EMcal}} - \phi_{\text{INTT tracklet}}$$

Angle resolution:

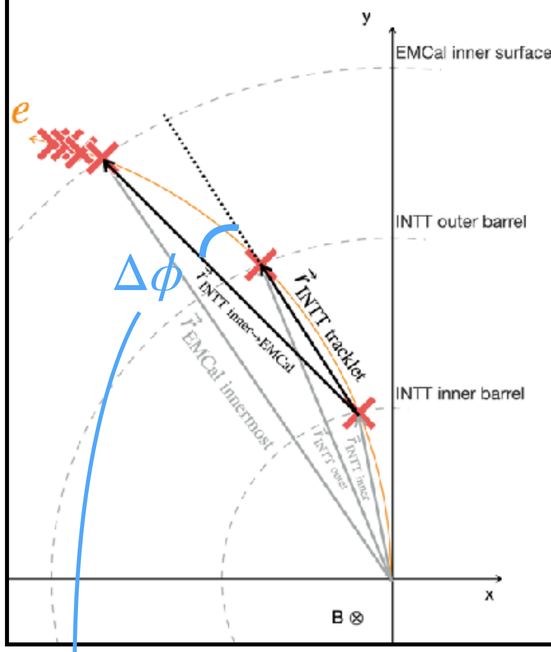
$$\frac{\sigma_{\Delta\phi}}{\left|\mu_{\Delta\phi}\right|},$$

where μ and σ are determined by gaussian fit to $\Delta\phi$ distribution

p_T vs $\Delta\phi$



Basic idea of angular resolution evaluation



$$\vec{r} = \vec{r}_{\text{mes.}} - \vec{r}_{\text{vertex}}\Big|_{\vec{r}_{\text{vertex}}=(0,0,0)} = \vec{r}_{\text{mes.}}$$

EMCal hit: the innermost (smallest ρ) G4Hit on EMCal

$$\vec{r}_{\text{EMCal innermost}}$$

INTT tracklet vector: a line from G4Hit on the inner barrel to that on the outer

$$\vec{r}_{\text{INTT tracklet}} = \vec{r}_{\text{INTT outer}} - \vec{r}_{\text{INTT inner}}$$

A vector from INTT inner barrel to EMcal hit:

$$\vec{r}_{\text{inner}\to\text{EMCal}} = \vec{r}_{\text{EMCal}} - \vec{r}_{\text{inner}}$$

Angle resolution evaluator:

$$\Delta\phi = \phi_{\text{inner}\to\text{EMcal}} - \phi_{\text{INTT tracklet}}$$

Angle resolution:

$$\frac{\sigma_{\Delta\phi}}{\left|\mu_{\Delta\phi}\right|},$$

where μ and σ are determined by gaussian fit to $\Delta\phi$ distribution

$$\Delta\phi = \Delta\phi(p_T, \phi, \eta, z, \text{PID})$$

$$\rightarrow \Delta\phi(p_T) = \frac{p_0}{p_T}$$

Is this function correct to use?

$$p_0 = -0.1922 \pm 0.003545$$

$$p_T(\Delta\phi) = \frac{p_0}{\Delta\phi}$$

C++ functions to provide $p_T(\Delta\phi)$ using $\Delta\phi(p_T)$

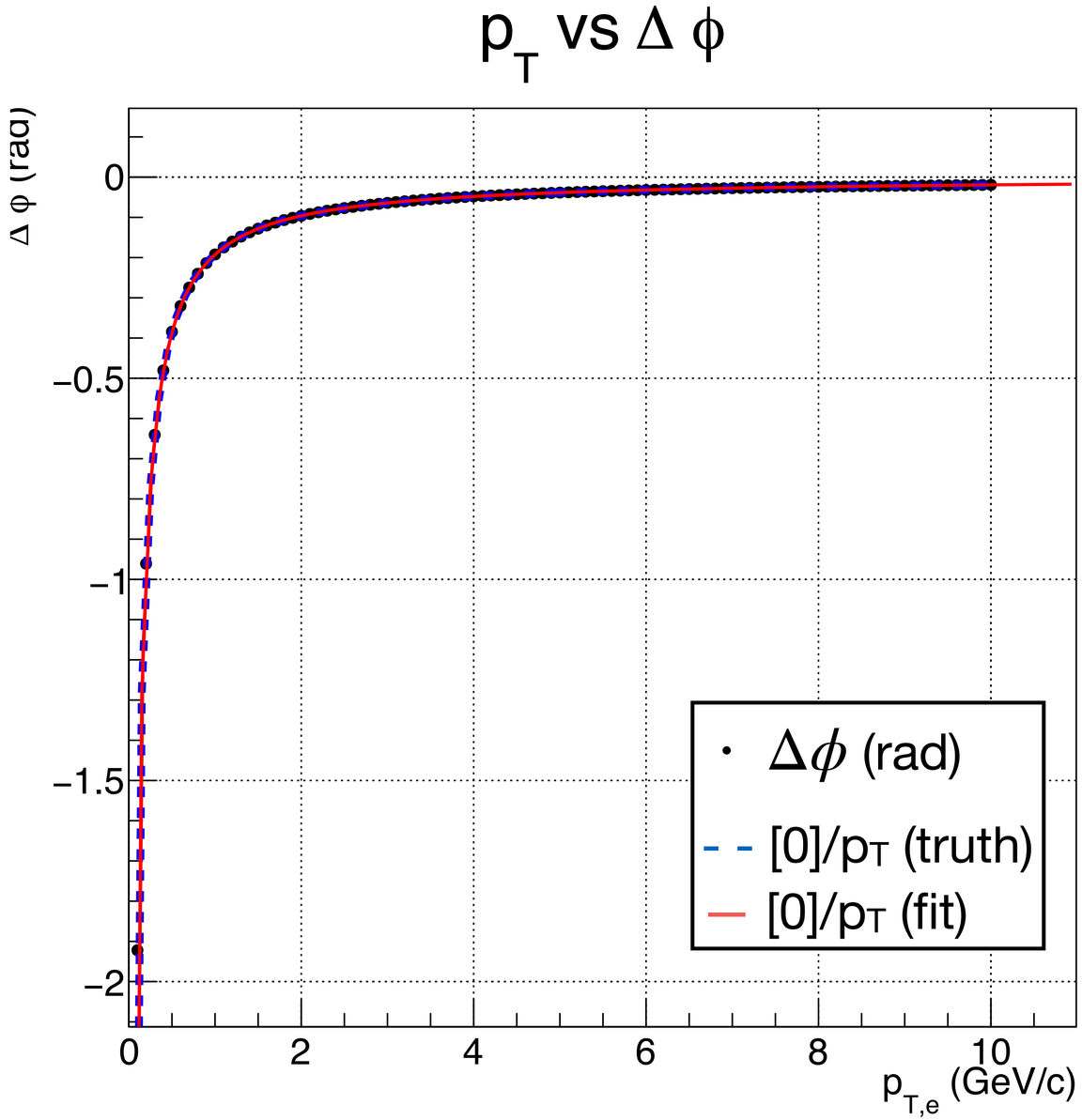
Header

```cpp
#pragma once

namespace SiCalo
{
  const double pt_dphi_par0 = -0.1922;

  //! Pt of electron in GeV/c is calculated using given dphi (rad), which is a difference of phi at INTT and ECal, at EMCal surface
  double GetPt( double dphi );

  //! Phi of electron in radian in [0, 2pi] is calculated using givein pT in GeV/c.
  double GetDphi( double pt );

};


#ifdef __CINT__
#include "SiCalo_pt_ideal.cc"
#endif // __CINT__

```

Source

```cpp
#include "SiCalo_pt_ideal.hh"

namespace SiCalo
{

  //! Pt of electron in GeV/c is calculated using given dphi (rad), which is a difference of phi at INTT and ECal, at EMCal surface
  double GetPt( double dphi )
  {
    //return fabs(pt_dphi_par1 / ( dphi - pt_dphi_par0 ) );
    return pt_dphi_par0 / dphi ;
  }

  //! Phi of electron in radian in [0, 2pi] is calculated using givein pT in GeV/c.
  double GetDphi( double pt )
  {

  return pt_dphi_par0 / pt;
  }

};
```

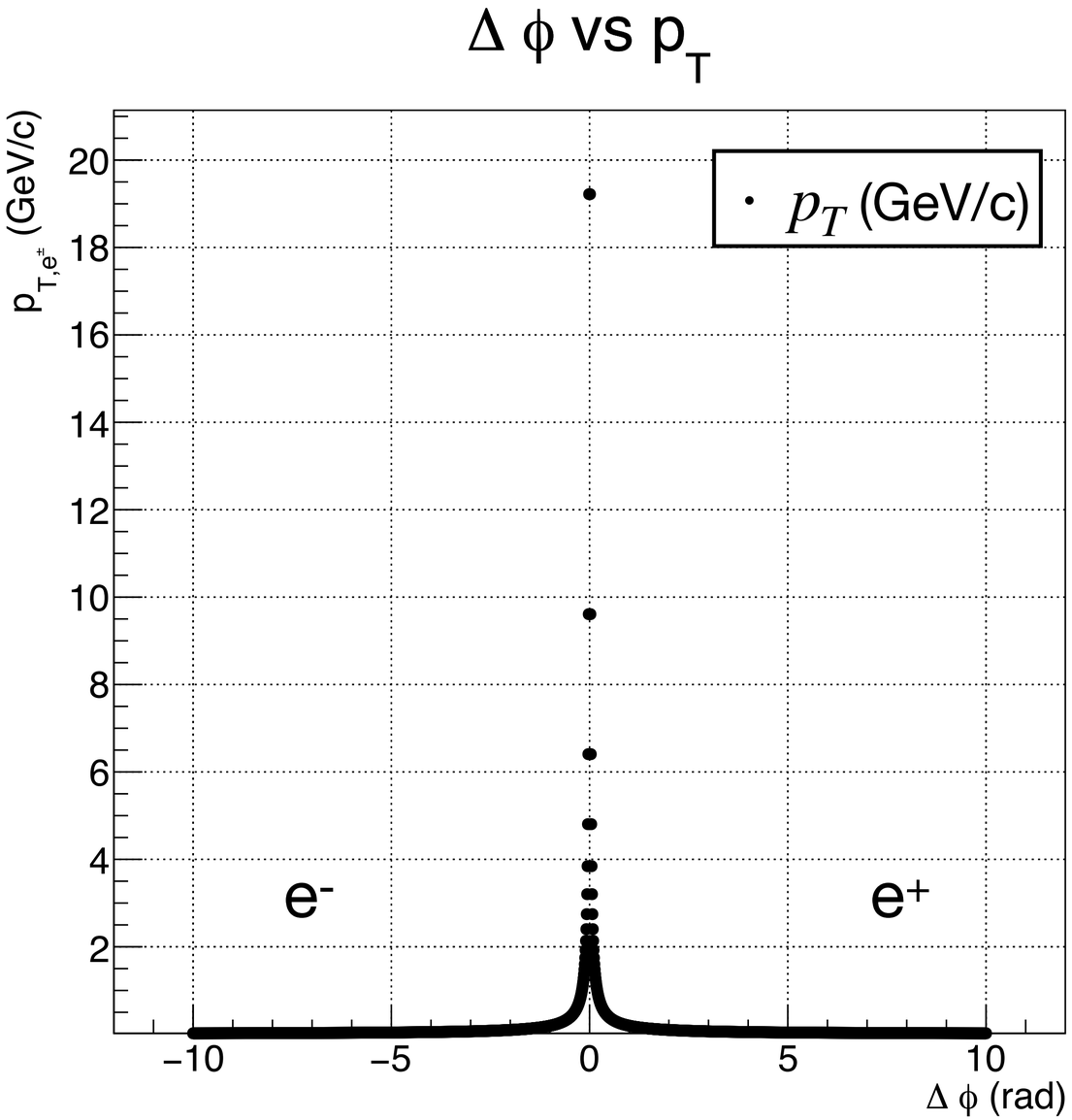
C++ functions to provide p_T using $\Delta\phi$



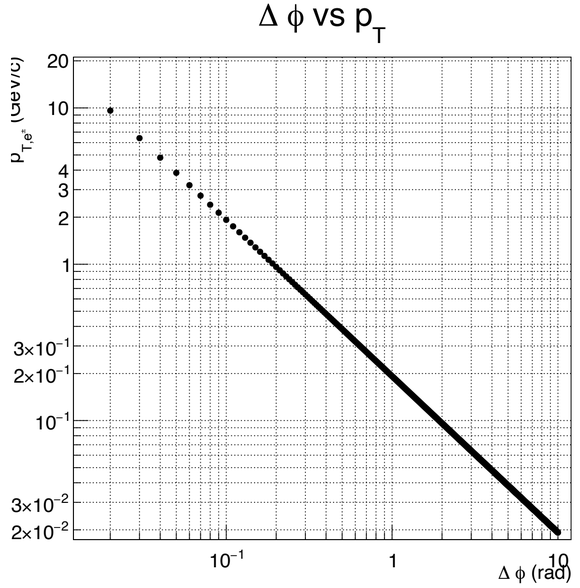

```
33  ////////////////////////////////////////////////
34  TGraph* g_pt_dphi = new TGraph();
35  g_pt_dphi->SetTitle( "p_{T} vs #Delta #phi; p_{T,e} (GeV/c);#Delta #phi (rad)" );
36  g_pt_dphi->SetMarkerStyle( 20 );
37  double dpt = 0.1; // GeV/c
38  double pt = 0.0;
39  for( int i = 0; i<100; i++ )
40    {
41      pt += dpt;
42      g_pt_dphi->AddPoint( pt, SiCalo::GetDphi( pt ) );
43    }
44
45  g_pt_dphi->Draw( "AP" );
46
47  TF1* f = new TF1( "f", "[0] /x", 0, 10 );
48  f->SetLineColor( kBlue );
49  f->SetLineStyle( 7 );
50  f->SetLineWidth( 5 );
51  f->SetParameters( -0.1922 );
52  f->Draw( "sameL" );
53
54  TF1* f_fit = new TF1( "f", "[0]/x", 0, 10 );
55  g_pt_dphi->Fit( f_fit );
56  f_fit->Draw( "sameL" );
```

```
Processing macro/SiCalo_pt_ideal.cc...
Info in <TCanvas::Print>: pdf file results/SiCalo/demo.pdf has been created
****************************************
Minimizer is Minuit2 / Migrad
Chi2                      =  2.20142e-19
NDf                       =           99
Edm                       =  2.20142e-19
NCalls                    =           13
p0                        =      -0.1922   +/-   3.68788e-12
```

C++ functions to provide $\Delta\phi$ using p_T




```
60  ////////////////////////////////////////////////
61  TGraph* g_dphi_pt = new TGraph();
62  g_dphi_pt->SetTitle( "#Delta #phi vs p_{T};#Delta #phi (rad); p_{T,e^{#pm}} (GeV/c)" );
63  g_dphi_pt->SetMarkerStyle( 20 );
64
65  double dphi = 10e-3; // GeV/c
66  double phi = 0.0;
67  for( int i = 0; i<100; i++ )
68    {
69      phi += dphi;
70      g_dphi_pt->AddPoint( phi, SiCalo::GetPt( phi ) );
71      g_dphi_pt->AddPoint( -phi, SiCalo::GetPt( -phi ) );
72    }
73
74  g_dphi_pt->Draw( "AP" );
75  c->Print( c->GetName() );
```



Where is it?

https://github.com/nukazuka/sPHENIX_SiCalo/tree/main