Minimum bias pp collisions using spdroot-4.1.7.6

System: p + p √s = 27 GeV

Statistics: 772 000 events

Версия SpdRoot	МК-генератор	Детекторы	Пучок
spdroot-4.1.7.6	Pythia8	DSSD (вершинный детектор), Straw Tracker, TOF, ECAL, Range System, BBC, ZDC, FARICH	Размытие по Гауссу σ = 1 мм в плоскости XY σ = 30 см по оси Z

PROD2026-005

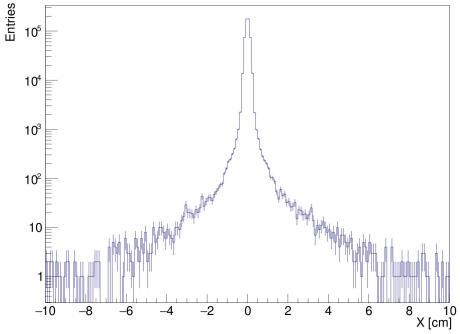
Положение первичной вершины

p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

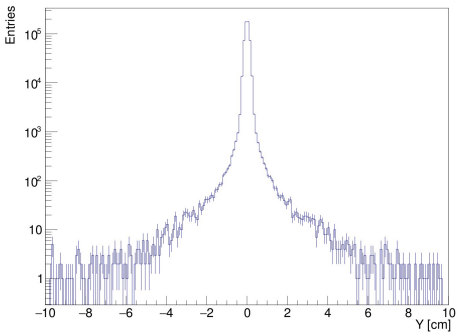
Размытие пучка по Гауссу
SpdVertexRC

σ = 1 мм в плоскости XY
σ = 30 см по Z

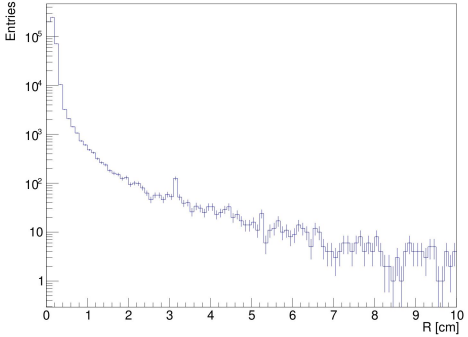
Восстановленная координата X первичной вершины



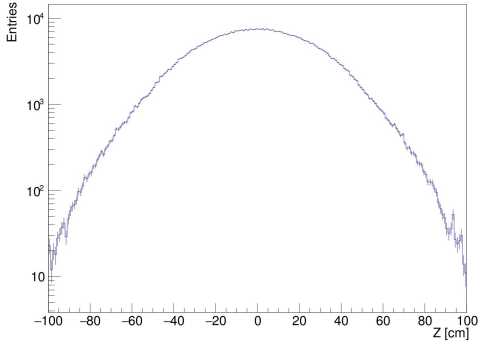
Восстановленная координата Y первичной вершины



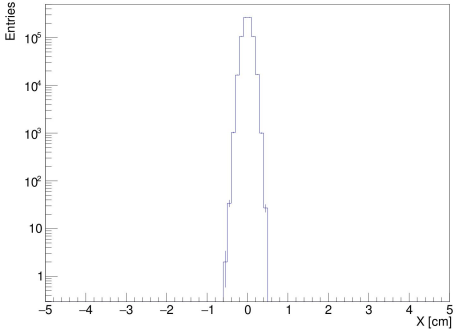
Vertex R = $\sqrt{X^2+Y^2}$



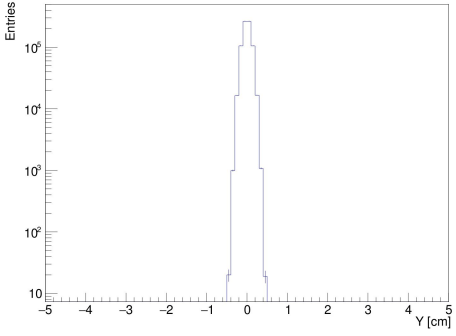
Восстановленная координата Z первичной вершины



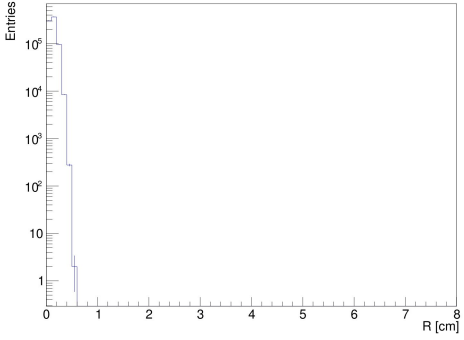
Координата X первичной вершины из Geant4



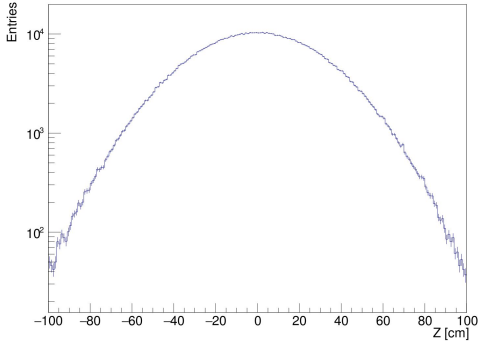
Координата Y первичной вершины из Geant4



Vertex R = $\sqrt{X^2+Y^2}$



Координата Z первичной вершины из Geant4



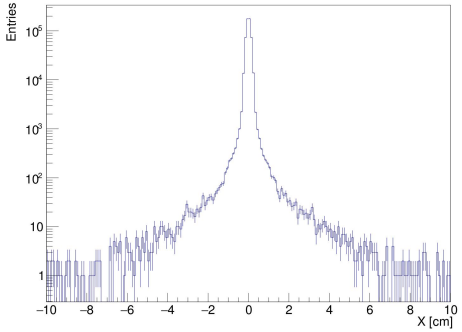
Положение первичной вершины

p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

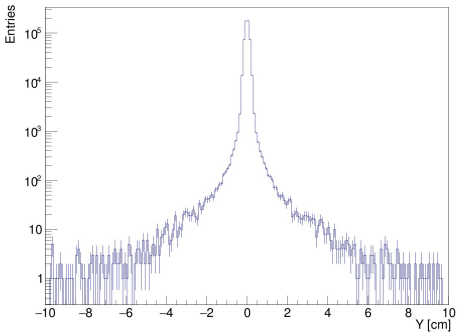
Размытие пучка по Гауссу
SpdVertexRC

σ = 1 мм в плоскости XY
σ = 30 см по Z

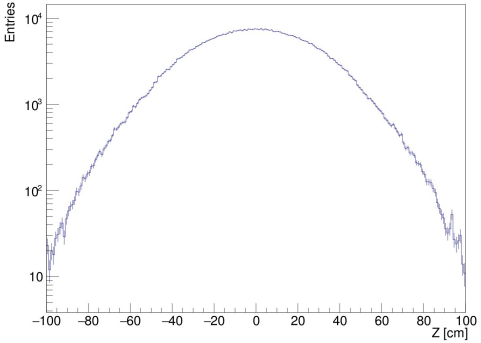
Восстановленная координата X первичной вершины



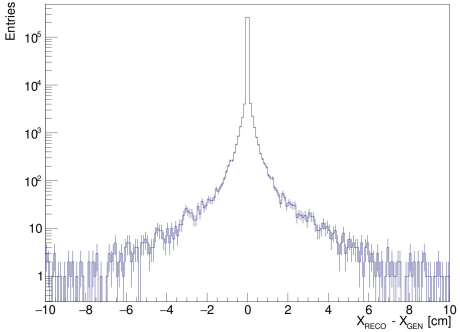
Восстановленная координата Y первичной вершины



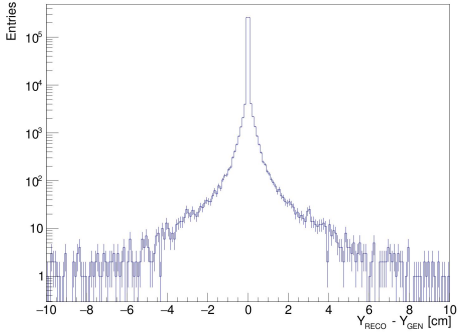
Восстановленная координата Z первичной вершины



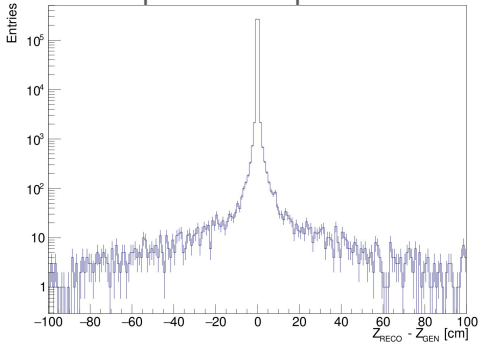
Разность между восстановленной и истинной (Geant4) координатами X первичной вершины



Разность между восстановленной и истинной (Geant4) координатами Y первичной вершины



Разность между восстановленной и истинной (Geant4) координатами Z первичной вершины



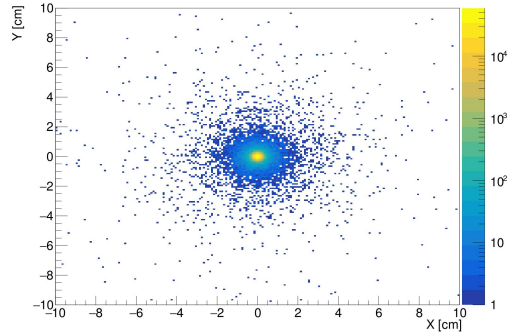
Положение первичной вершины

p + p √s = 27 ГэВ, spdroot-4.1.7.6

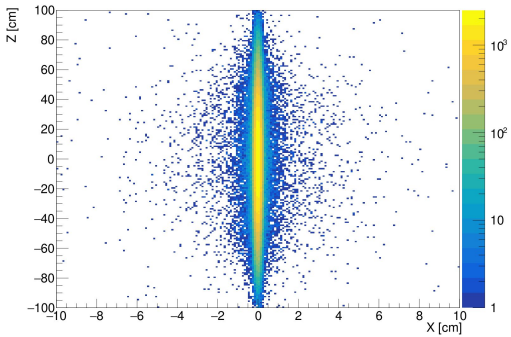
Размытие пучка по Гауссу
SpdVertexRC

σ = 1 мм в плоскости XY
σ = 30 см по Z

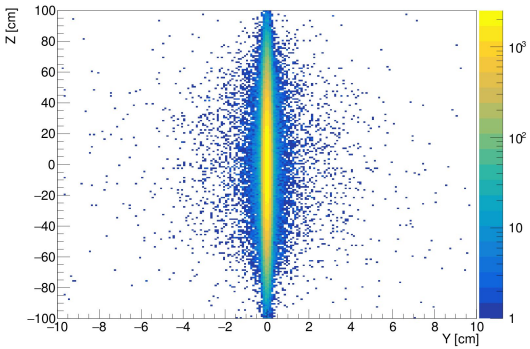
Восстановленное положение первичной вершины в плоскости XY



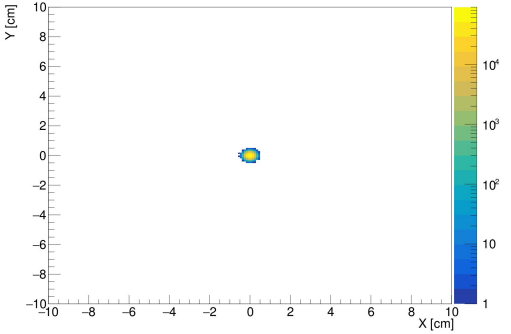
Восстановленное положение первичной вершины в плоскости XZ



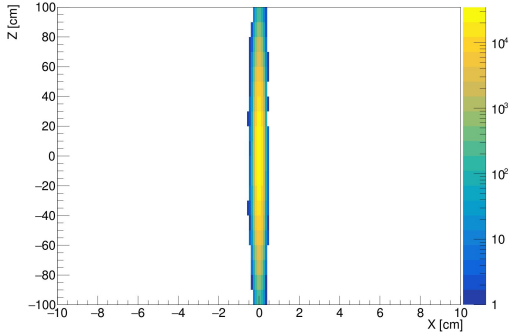
Восстановленное положение первичной вершины в плоскости YZ



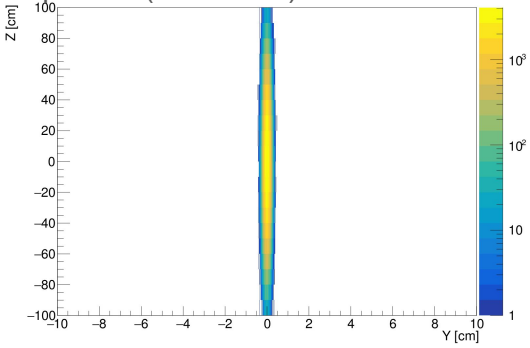
Истинное положение первичной вершины (из Geant4) в плоскости XY



Истинное положение первичной вершины (из Geant4) в плоскости XZ



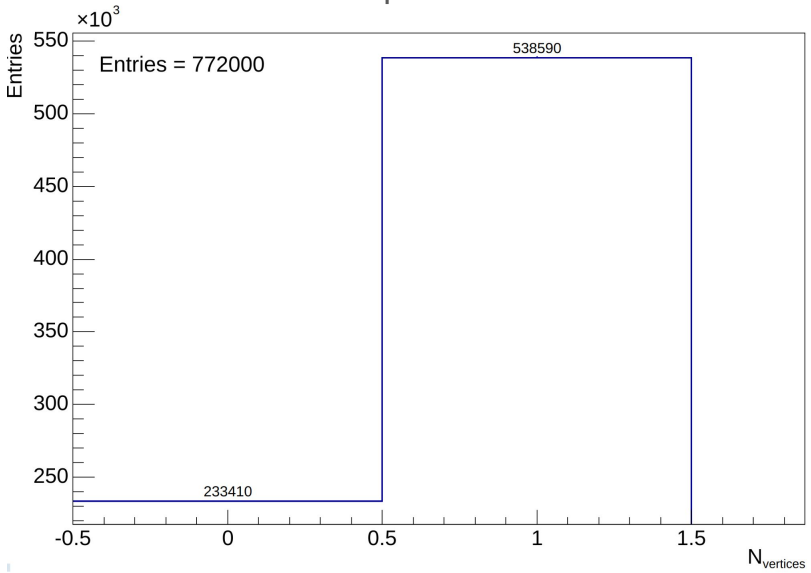
Истинное положение первичной вершины (из Geant4) в плоскости YZ



Восстановление первичной вершины (с помощью SpdRCVerticesFinder)

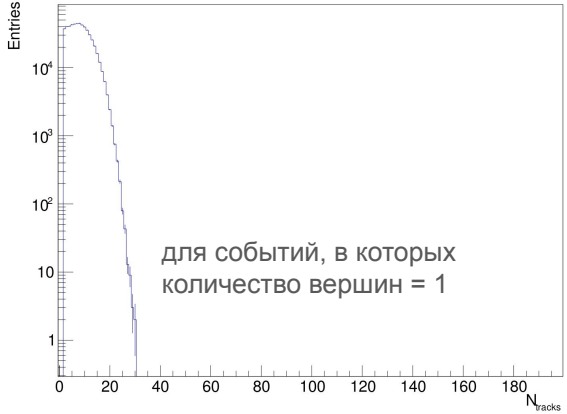
p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

Количество вершин в событии

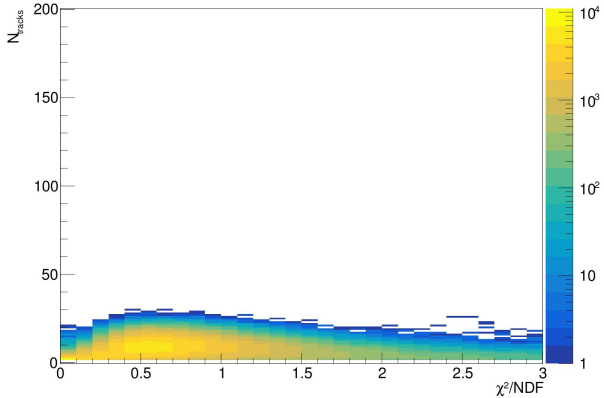


Первичная вершина не была восстановлена в 30% событий

Количество МС-треков, по которым восстановлена первичная вершина вершина



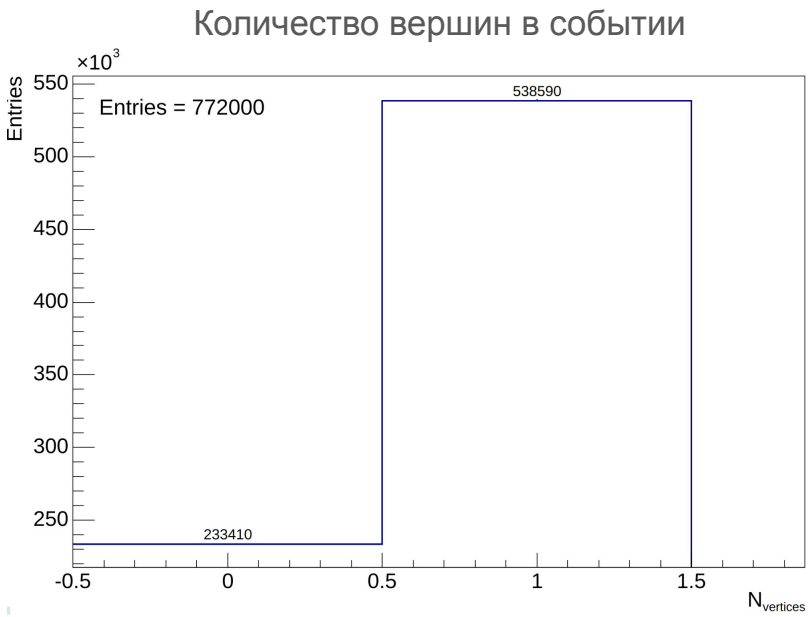
Vertex N_{tracks} vs Vertex χ^2/NDF

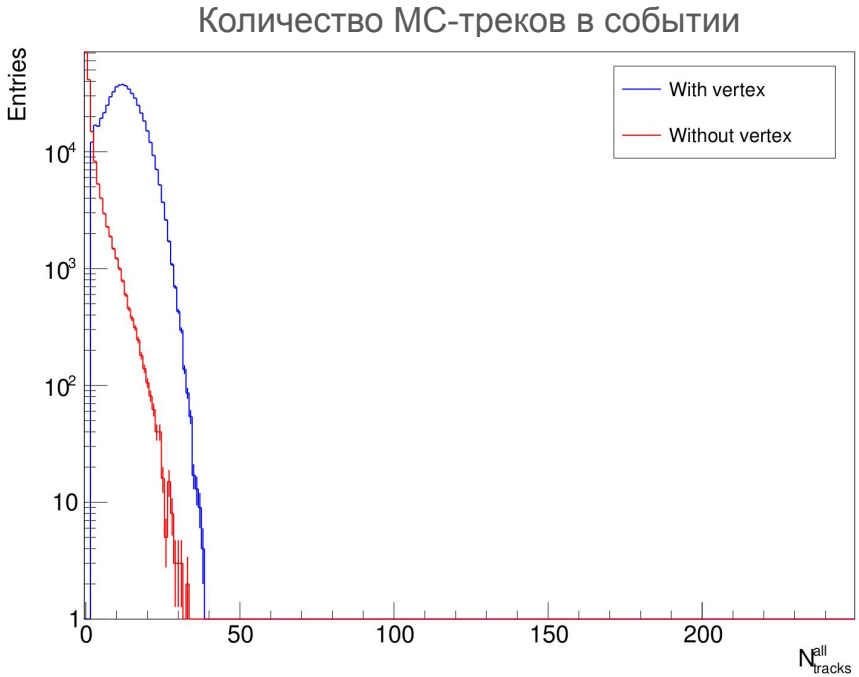


- Восстановление вершины происходит по МС-трекам
- МС-треки – восстановленные треки с использованием информации из генератора

Восстановление первичной вершины (с помощью SpdRCVerticesFinder)

p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6





Первичная вершина не была восстановлена в 30% событий

Все MC-треки без применения каких-либо критериев отбора

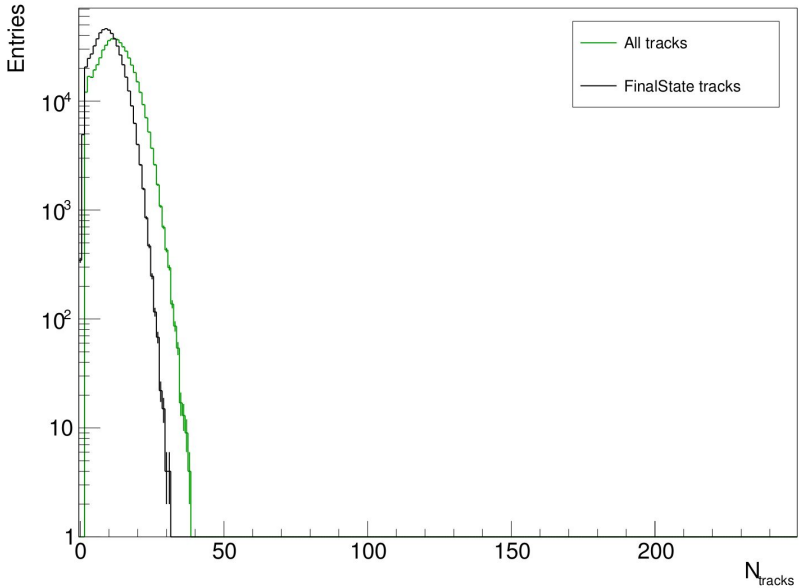
- Восстановление вершины происходит по MC-трекам
- MC-треки – восстановленные треки с использованием информации из генератора

Восстановление первичной вершины (с помощью SpdRCVerticesFinder)

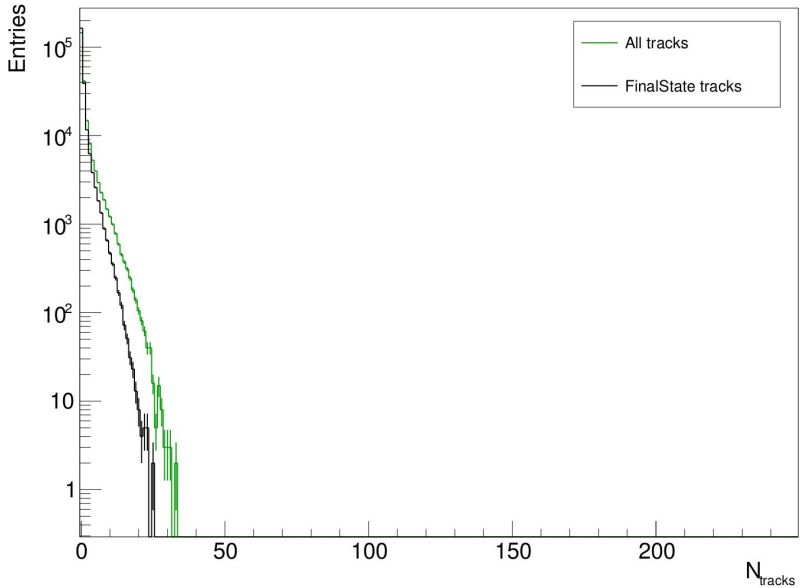
p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

- В событиях, в которых первичная вершина не была восстановлена имеются FinalState треки
- FinalState – state in the primary vertex if it's found

Распределение для событий, в которых **вершина восстановлена**



Распределение для событий, в которых **вершина не восстановлена**



Положение треков в первичной вершине

p + p √s = 27 ГэВ, spdroot-4.1.7.6

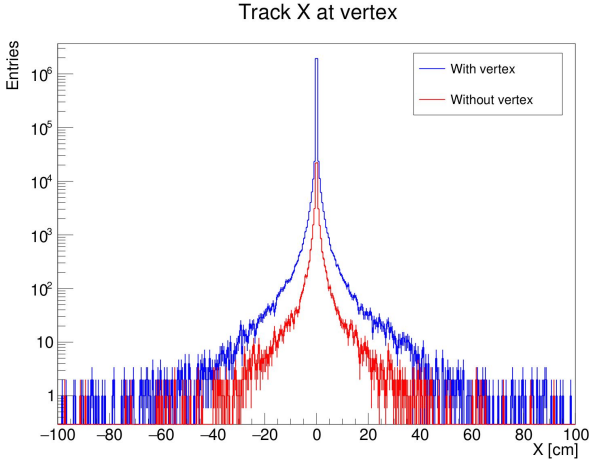
В событиях, в которых первичная вершина не была восстановлена имеются FinalState треки

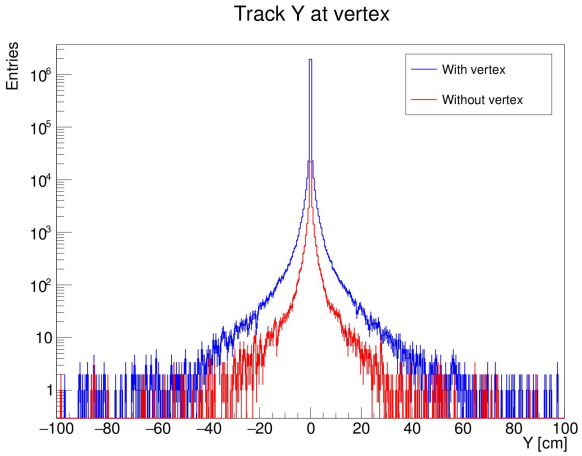
FinalState – state in the primary vertex if it's found

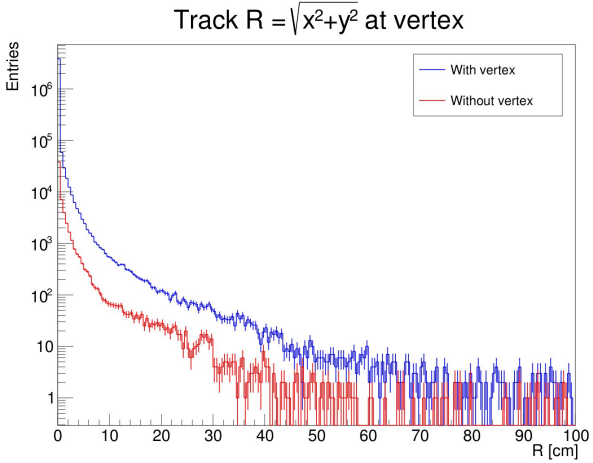
FinalState у трека создается тогда же, когда выполняется фит первичной вершины в коде SpdMCVerticesFitter

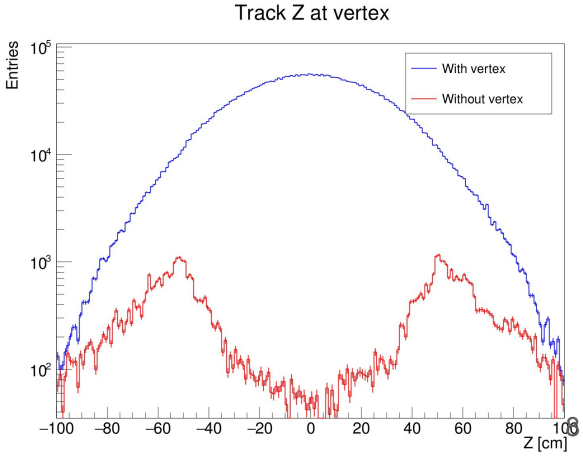
mc-vertex есть во всех событиях ⇒ в каждом событии выполняется попытка экстраполяции трека (и создания FinalState). Причем судя по всему экстраполяция выполняется в mc-vertex, а не в rc-vertex.

Но используем мы вершину, восстановленную в SpdRCVertexFinder









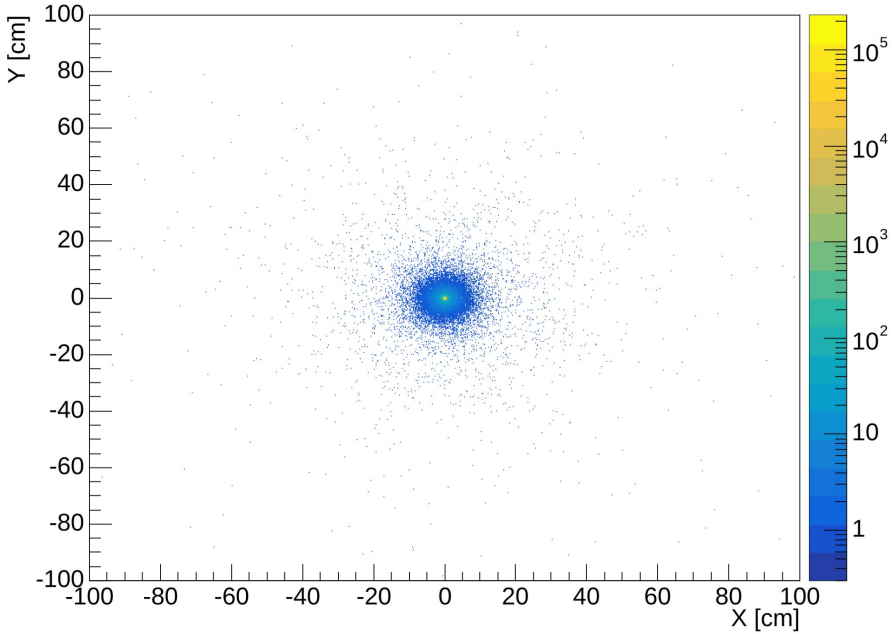
Положение треков в плоскости XY в первичной вершине

p + p √s = 27 ГэВ, spdroot-4.1.7.6

Размытие пучка по Гауссу:
σ = 1 мм в плоск. XY,
σ = 30 см по оси Z

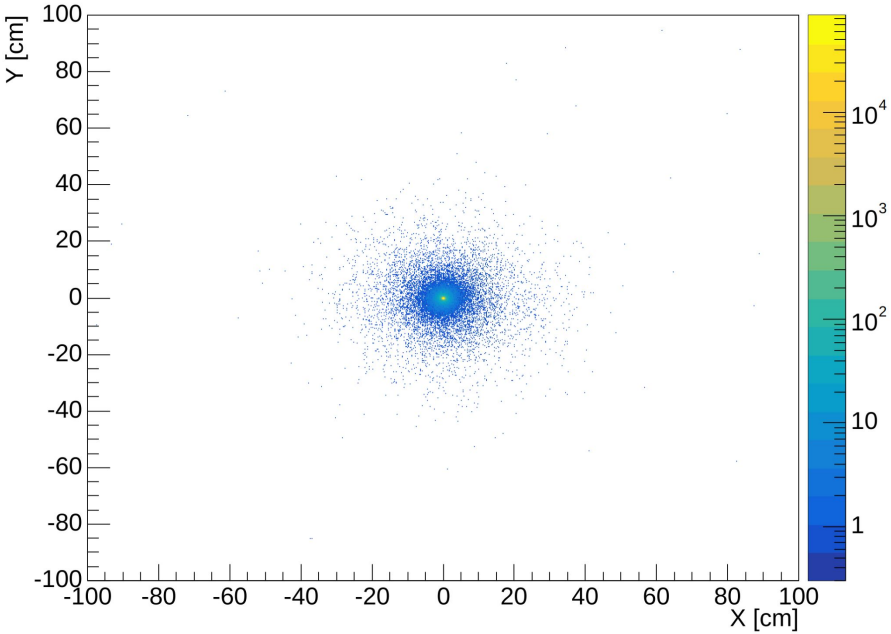
- События с восстановленной первичной вершиной
- SpdTrackMC: Final State

Barrel Track Y vs X (FinalState)



- Отобраны треки, у которых хиты только в barrel

Endcaps Track Y vs X (FinalState)



- Отобраны треки, у которых хиты только в endcaps

Положение треков в плоскости XY в первой измеренной точке

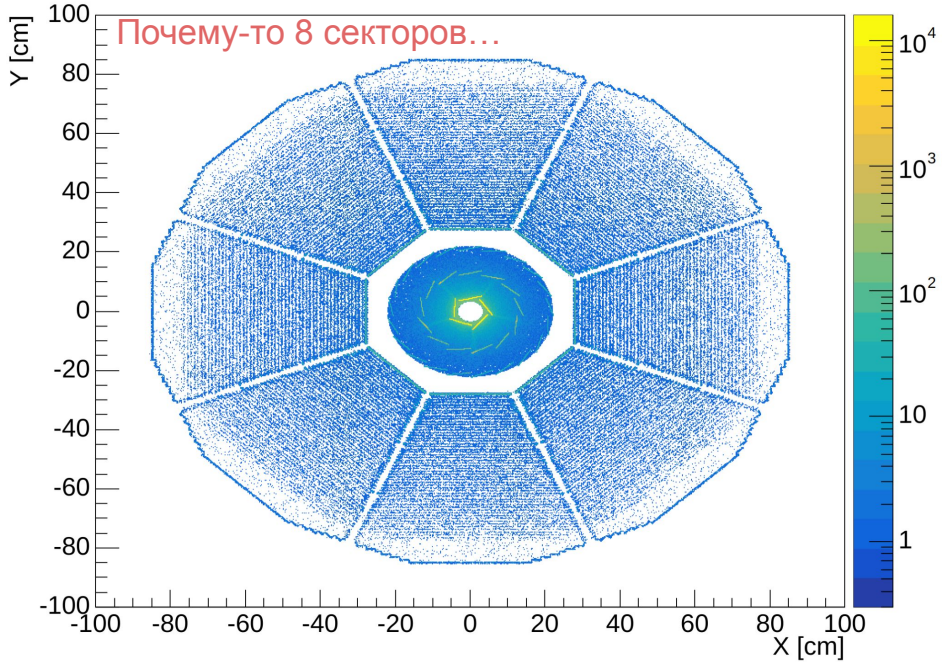
p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

- SpdTrackMC: First State

Размытие пучка по Гауссу:
σ = 1 мм в плоск. XY,
σ = 30 см по оси Z

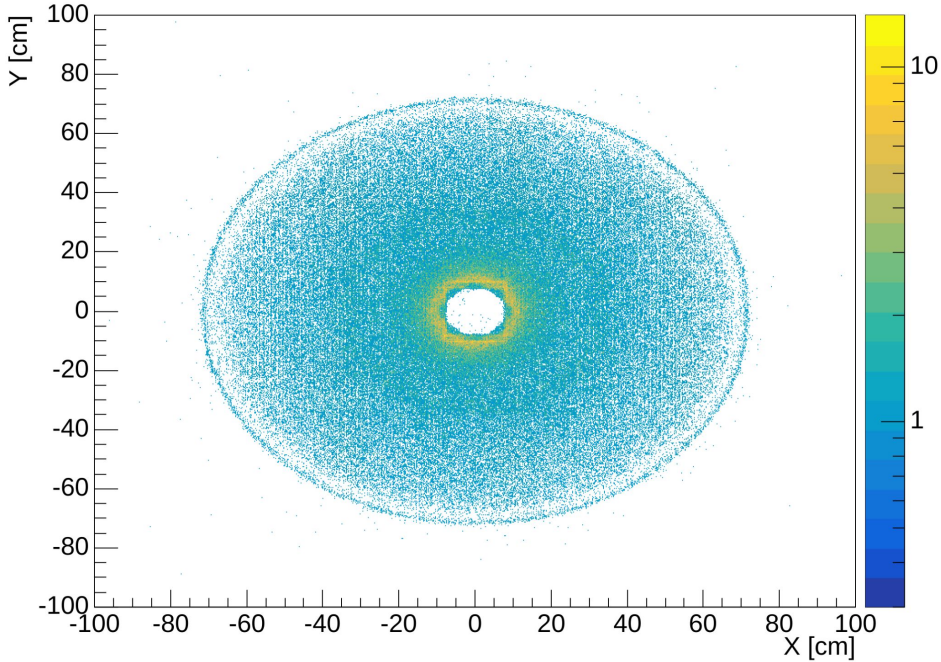
Straw Tracker Barrel

Track Y vs X (FirstState) |Z| < 120 cm



Straw Tracker Endcaps

Track Y vs X (FirstState) 125 < |Z| < 150 cm



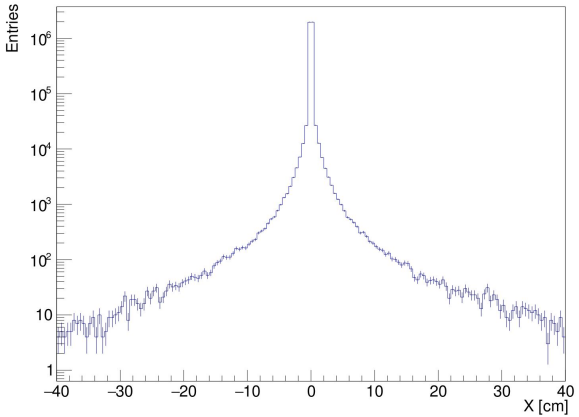
- **FirstState** – state in the first measured point

Разделение по координате Z трека

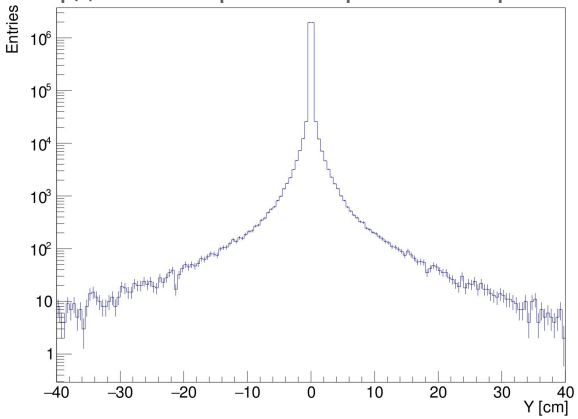
p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

- События с восстановленной первичной вершиной (SpdVertexRC)
- SpdTrackMC: Final State, isGood()

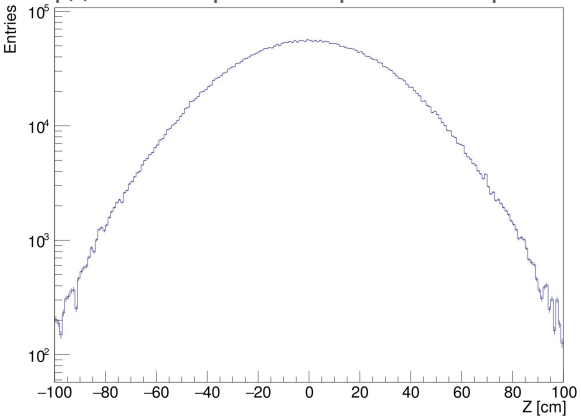
Координата X трека в первичной вершине



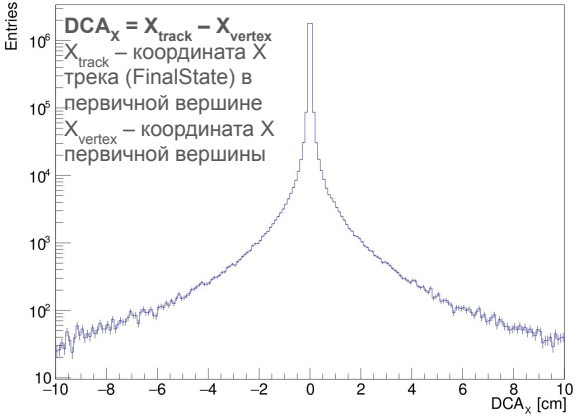
Координата Y трека в первичной вершине

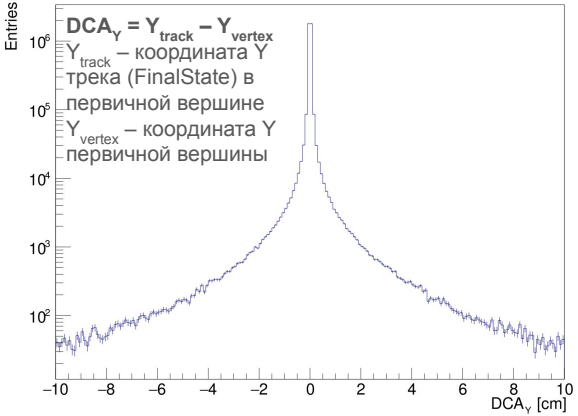


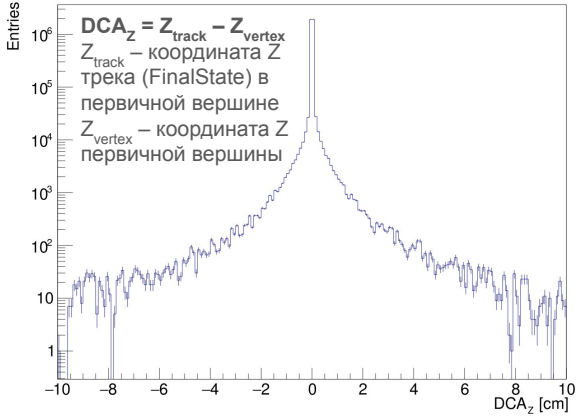
Координата Z трека в первичной вершине



Distance of closest approach (DCA) = Track position at Vertex (Final State) – Vertex position

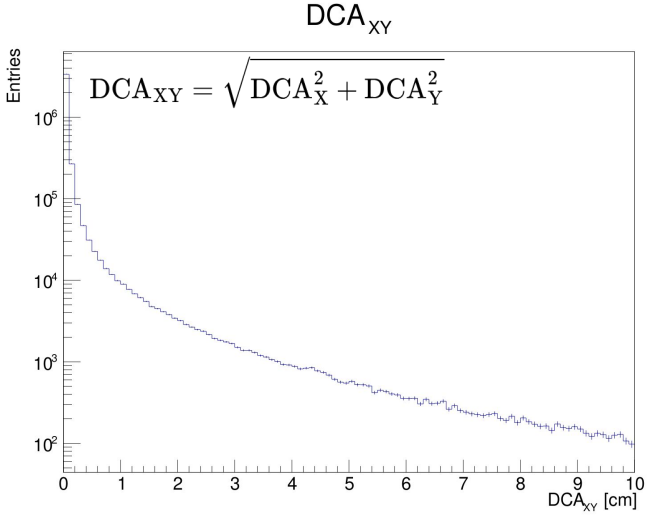


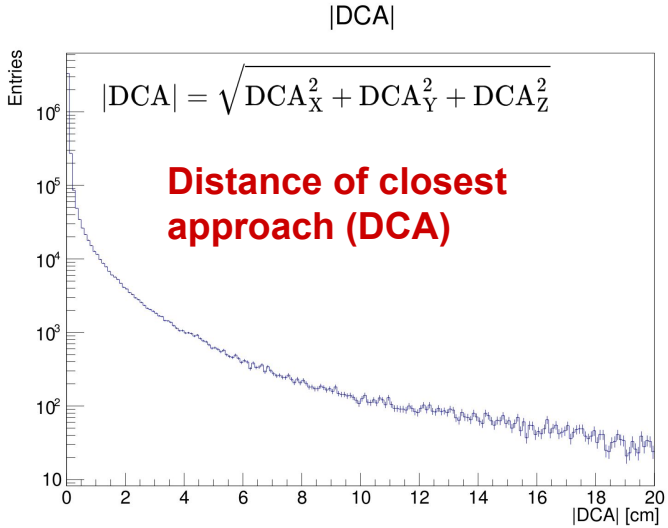


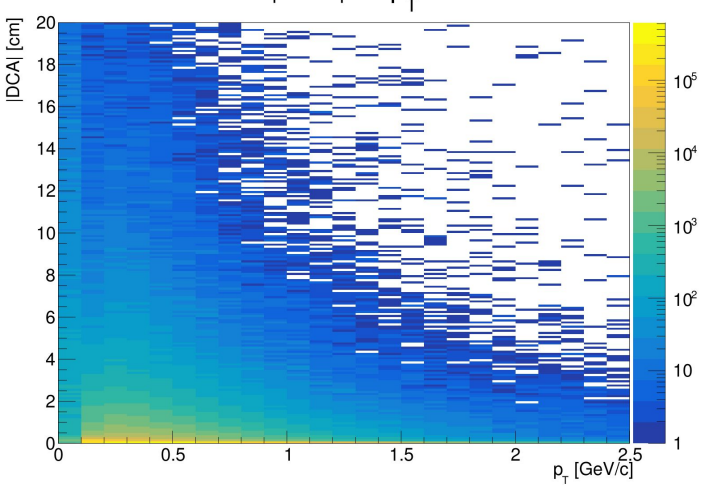


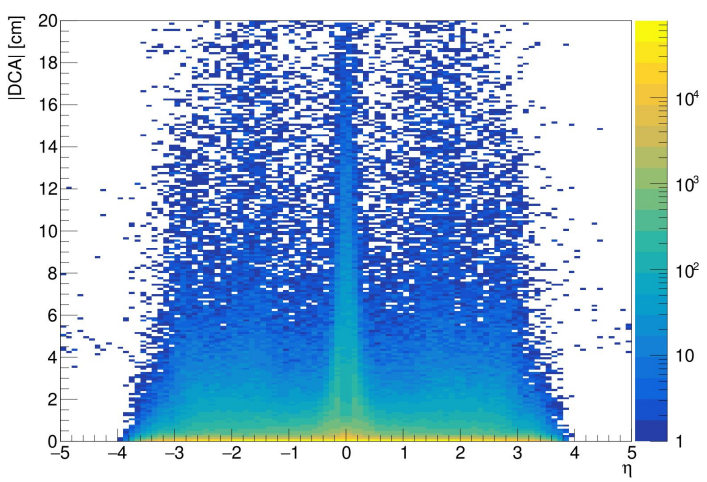
p + p √s = 27 ГэВ, spdroot-4.1.7.6

- События с восстановленной первичной вершиной
- SpdTrackMC: Final State, isGood()









SpdVertexMC vs. SpdVertexRC

p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

Размытие пучка по Гауссу σ = 1 мм в плоскости XY
σ = 30 см по Z

Решила я в итоге посмотреть различие между SpdVertexMC и SpdVertexRC.
Для понимания (если правильно разобралась в коде):
SpdVertexMC:

1) берется истинная вершина и выделяется набор частиц, которые в этой вершине родились.
2) отбираются треки, соответствующие этим частицам.
3) на треки накладывается ряд ограничений: трек сфитирован (track->GetIsFitted()), фит “хороший” (tpars->GetIsGood() – можно в бэкапе метод посмотреть), угол $|\theta - \pi/2| < 3$.
4) по оставшимся после отборов трекам определяется положение вершины.
5) треки экстраполируются в эту вершину и создается FinalState (положение трека в первичной вершине).

Я проверила, SpdVertexMC есть во всех событиях.

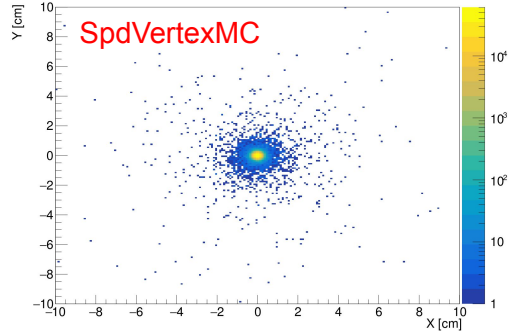
SpdVertexRC – берутся треки из события, на них накладываются все те же ограничения и выполняется восстановление вершины. Т.е. отличие в том, что теперь не отбираются треки только из этой вершины. Никакой экстраполяции не выполняется.

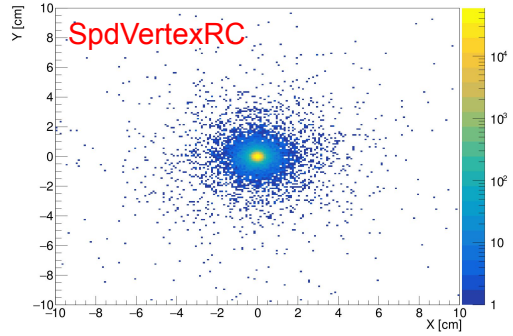
SpdVertexMC vs. SpdVertexRC

p + p √s = 27 ГэВ, spdroot-4.1.7.6

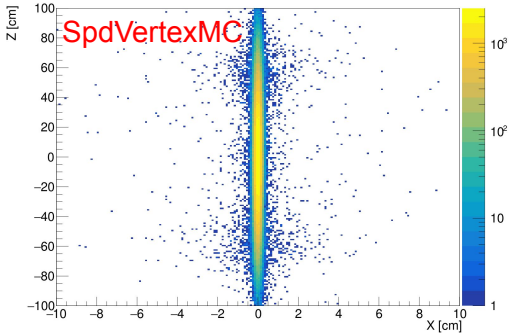
Размытие пучка по Гауссу σ = 1 мм в плоскости XY
σ = 30 см по Z

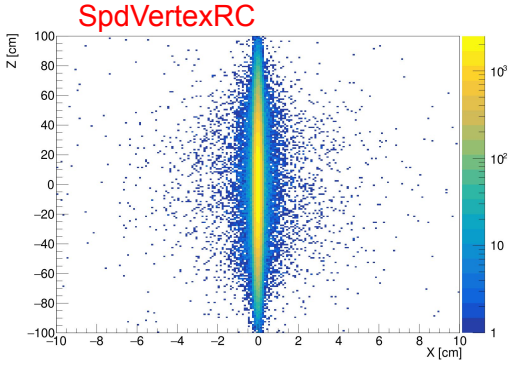
Восстановленное положение первичной вершины в плоскости XY



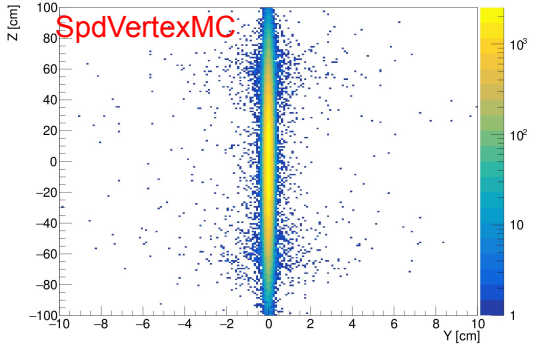


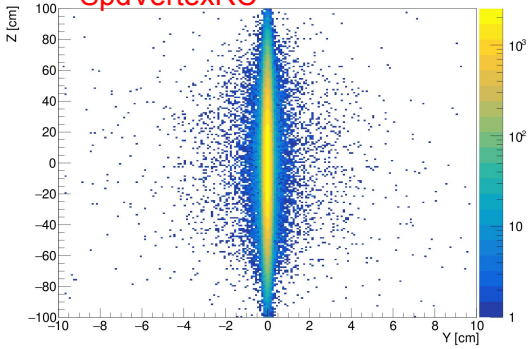
Восстановленное положение первичной вершины в плоскости XZ





Восстановленное положение первичной вершины в плоскости YZ





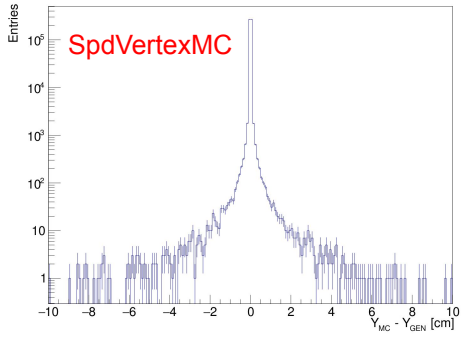
Распределение для SpdVertexMC менее размыто в плоскости XY

SpdVertexMC vs. SpdVertexRC

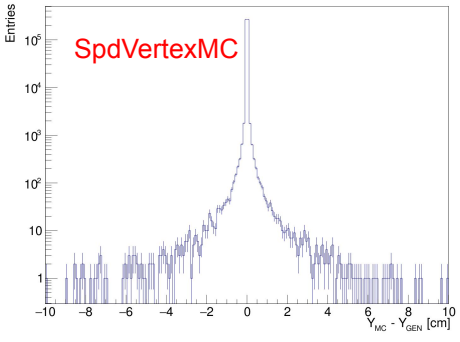
p + p √s = 27 ГэВ, spdroot-4.1.7.6

Размытие пучка по Гауссу σ = 1 мм в плоскости XY
σ = 30 см по Z

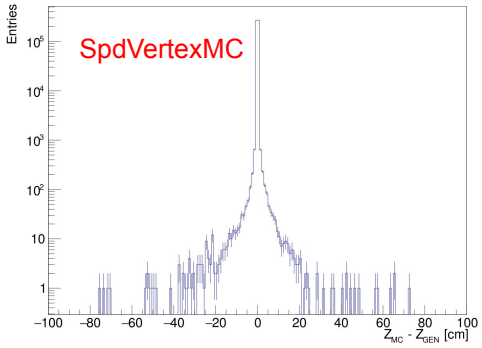
Разность между восстановленной и истинной (Geant4) координатами X первичной вершины



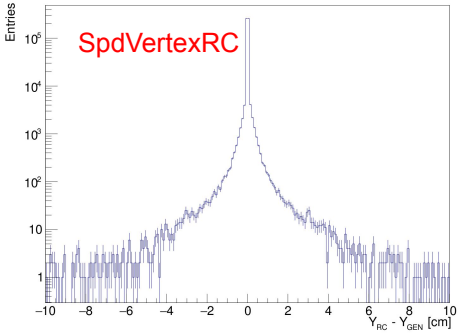
Разность между восстановленной и истинной (Geant4) координатами Y первичной вершины

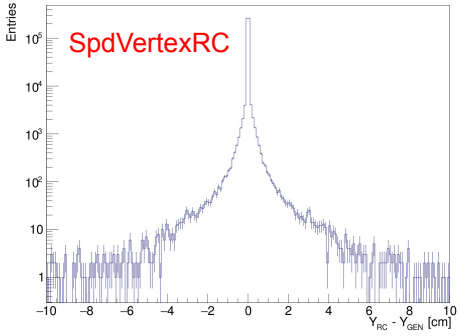


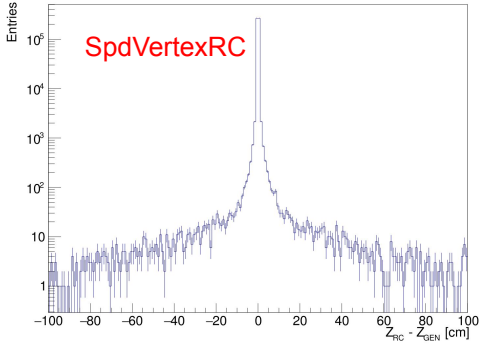
Разность между восстановленной и истинной (Geant4) координатами Z первичной вершины



Как будто значительных отличий нет, разве что по координате Z ситуация улучшилась



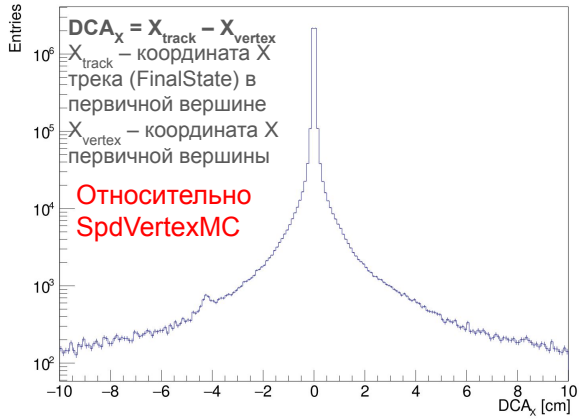


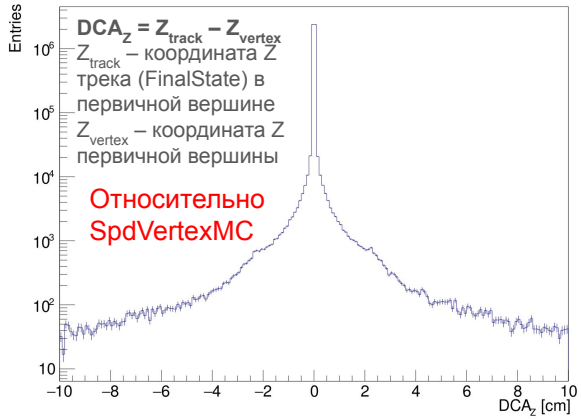


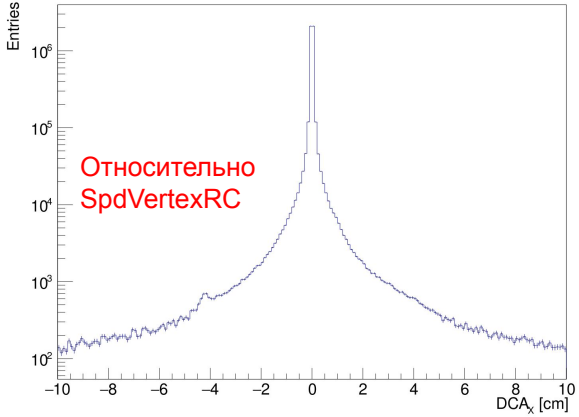
p + p $\sqrt{s}$ = 27 ГэВ, spdroot-4.1.7.6

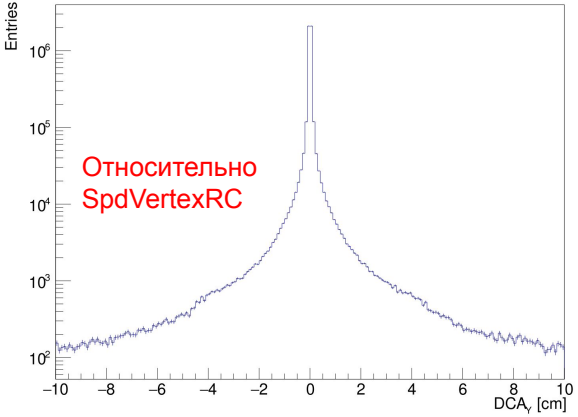
- События с восстановленной первичной вершиной
- SpdTrackMC: Final State

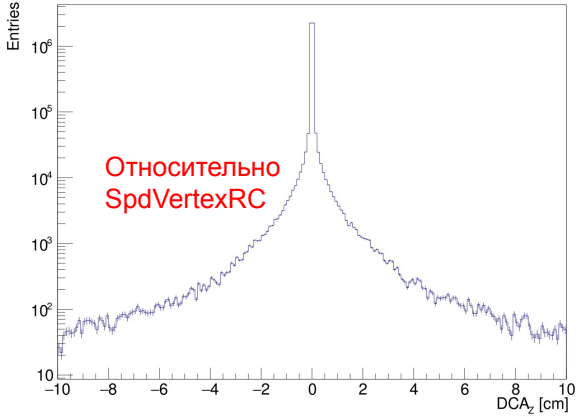
Distance of closest approach (DCA) = Track position at Vertex (Final State) – Vertex position







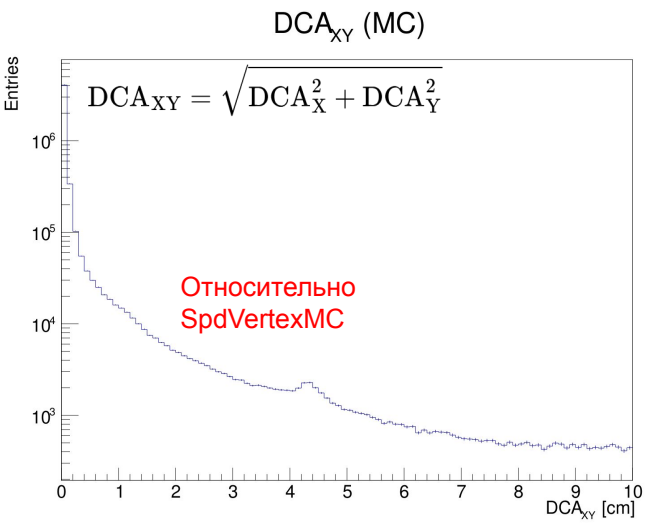


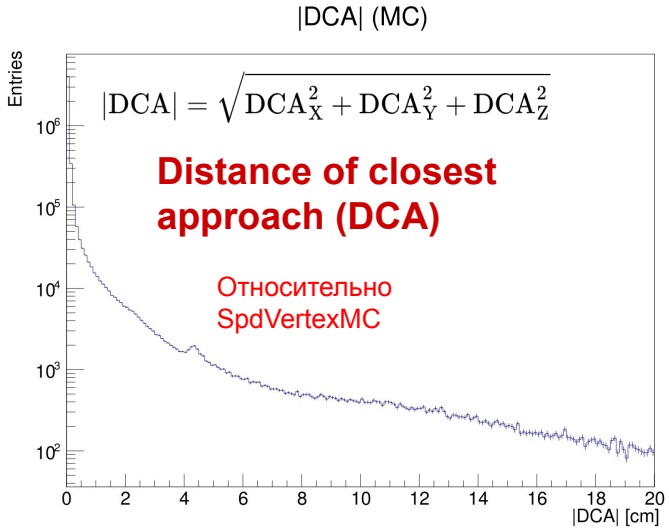


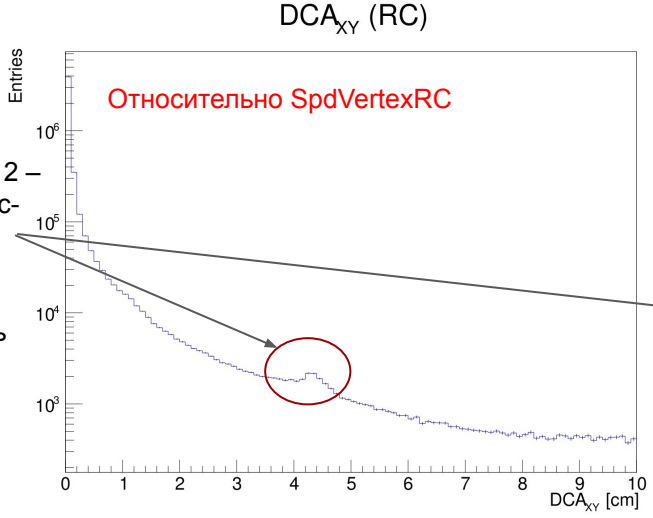
p + p √s = 27 ГэВ, spdroot-4.1.7.6

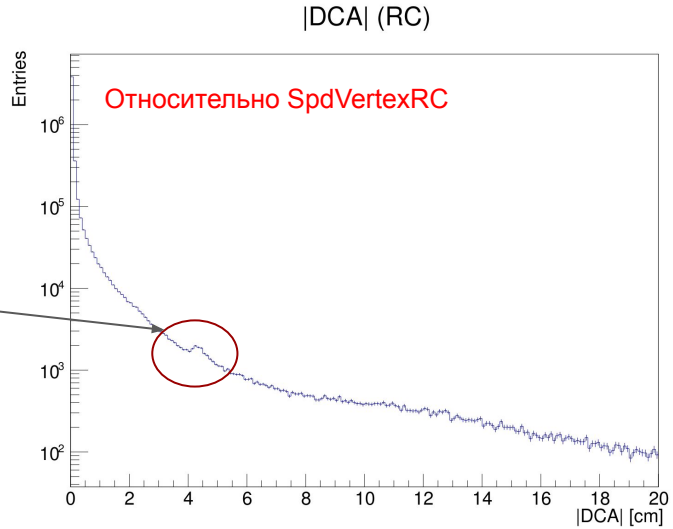
- События с восстановленной первичной вершиной
- SpdTrackMC: Final State

Если смотреть DCA относительно SpdVertexMC лучше как будто не становится.





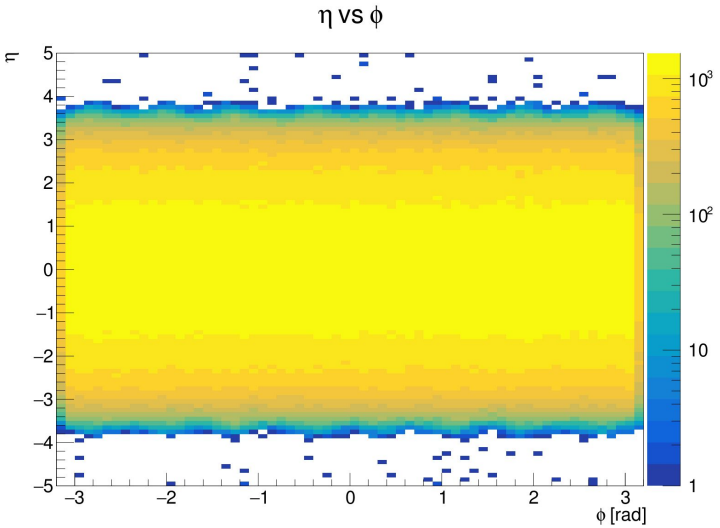




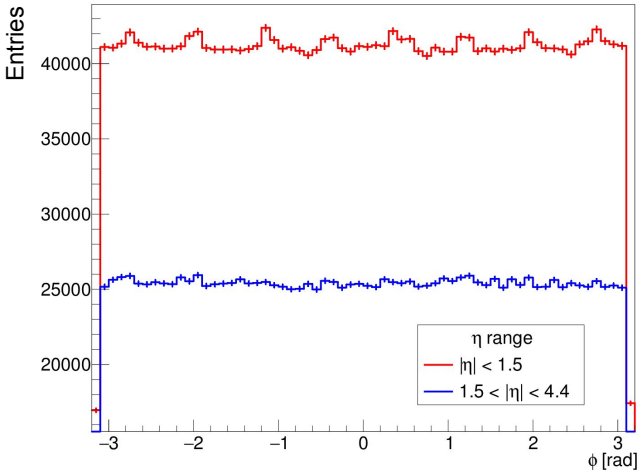
Кстати, если вернуться на сл. 12 – там этих горбиков для случая rc-вершин не было:
отличие в условии на fitPars.
Там я требую fitPars->isGood(), а здесь попробовала построить DCA без этого требования.

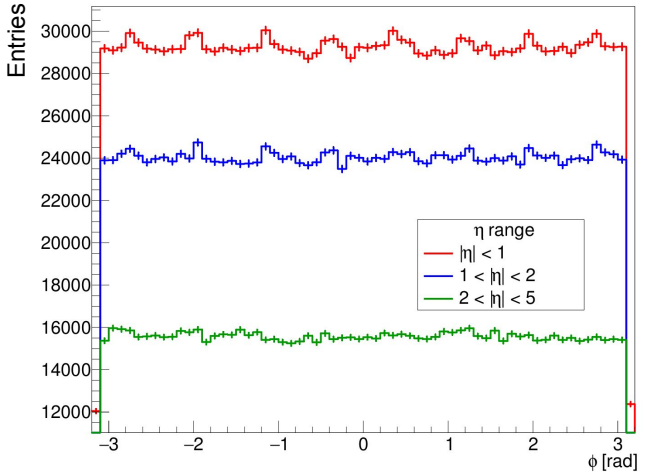
p + p √s = 27 ГэВ,
spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()



Распределения по азимутальному углу для разных диапазонов по псевдобыстроте



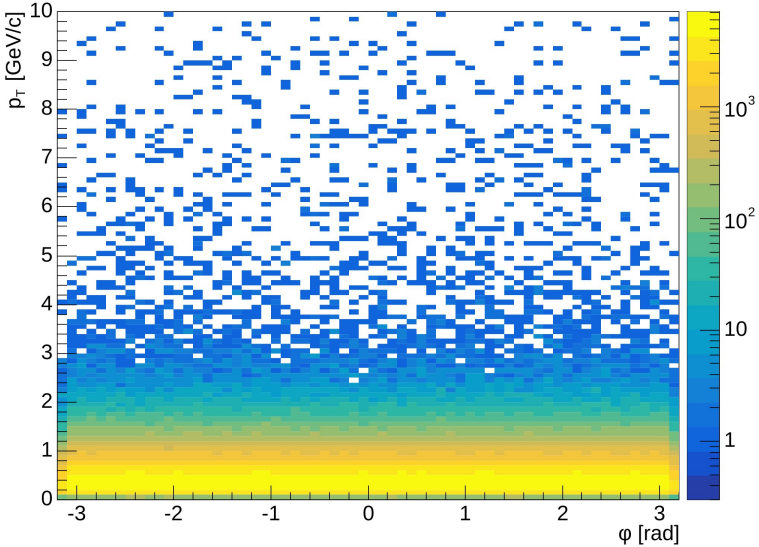


p + p √s = 27 ГэВ, spdroot-4.1.7.6

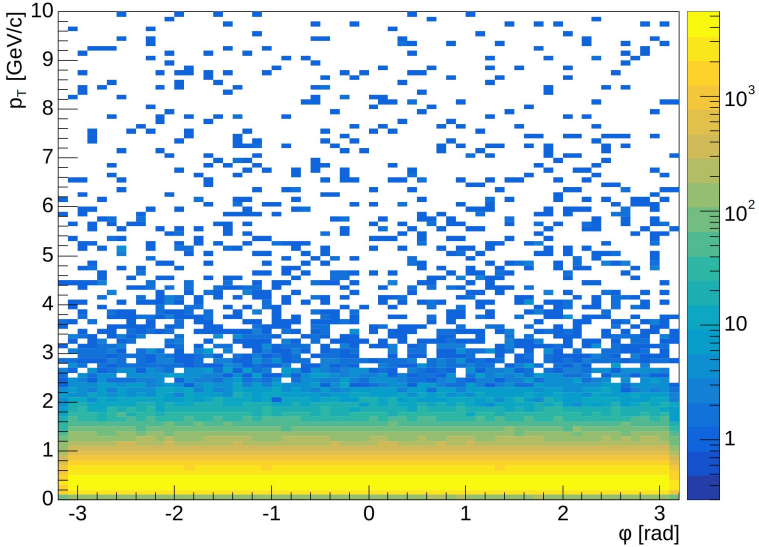
- SpdTrackMC: Final State, isGood()

- **FirstState** – state in the first measured point
- **LastState** – state in the last measured point
- **FinalState** – state in the primary vertex if it's found

p_T vs. φ
для положительно заряженных частиц



p_T vs. φ
для отрицательно заряженных частиц



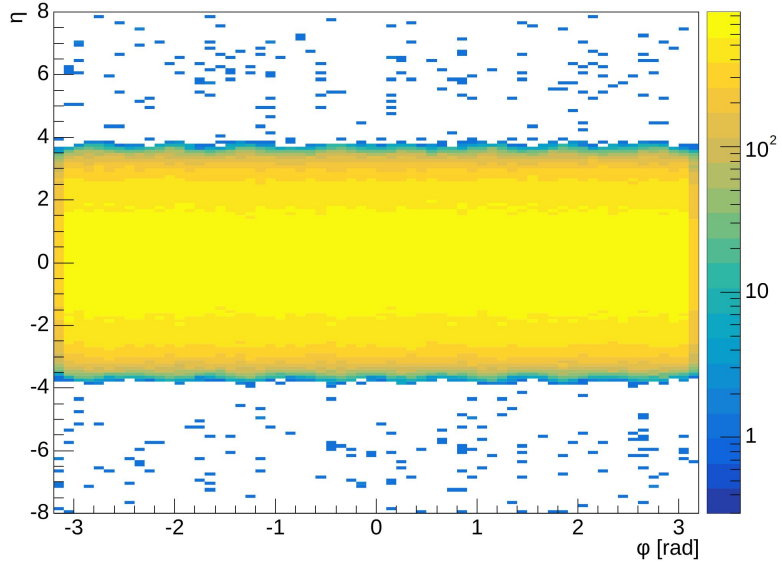
Отобраны треки, у которых есть FirstState, LastState, FinalState
Знак заряда определен по FirstState трека
Значения φ и p_T трека взяты из FinalState (т.е. состояния трека в первичной вершине)

p + p √s = 27 ГэВ, spdroot-4.1.7.6

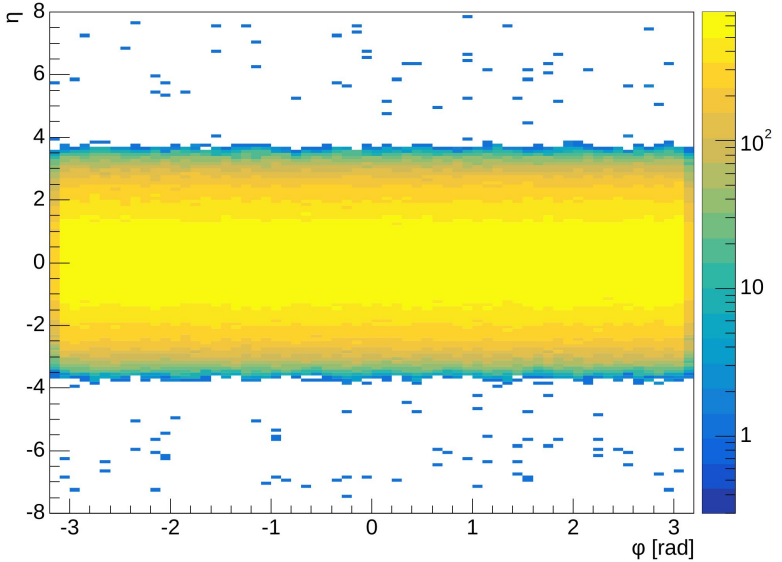
- SpdTrackMC: Final State, isGood()

- **FirstState** – state in the first measured point
- **LastState** – state in the last measured point
- **FinalState** – state in the primary vertex if it's found

η vs. φ
для положительно заряженных частиц



η vs. φ
для отрицательно заряженных частиц



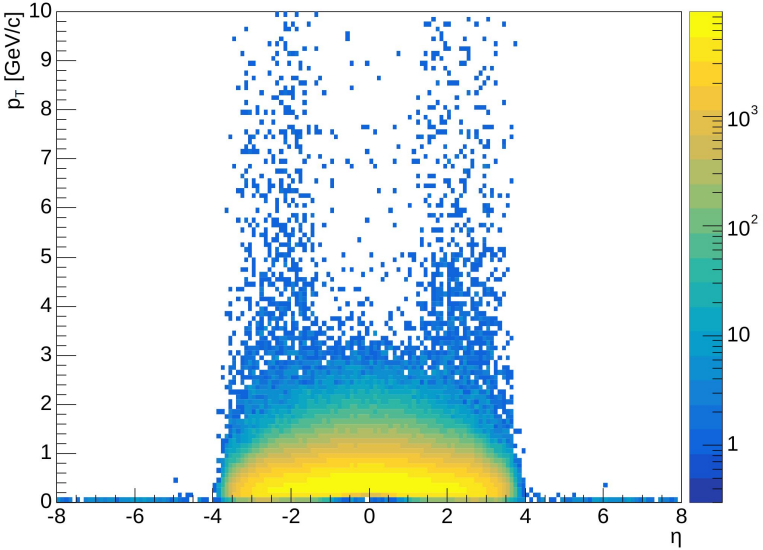
Отобраны треки, у которых есть FirstState, LastState, FinalState
Знак заряда определен по FirstState трека
Значения φ и η трека взяты из FinalState (т.е. состояния трека в первичной вершине)

p + p √s = 27 ГэВ, spdroot-4.1.7.6

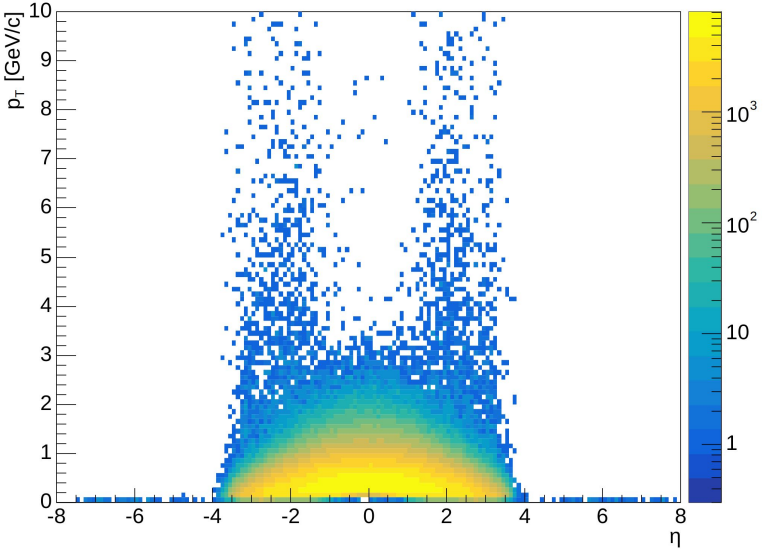
- SpdTrackMC: Final State, isGood()

- **FirstState** – state in the first measured point
- **LastState** – state in the last measured point
- **FinalState** – state in the primary vertex if it's found

p_T vs. η
для положительно заряженных частиц



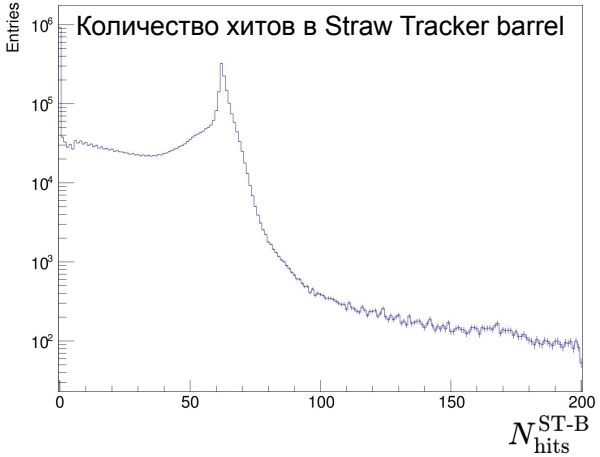
p_T vs. η
для отрицательно заряженных частиц

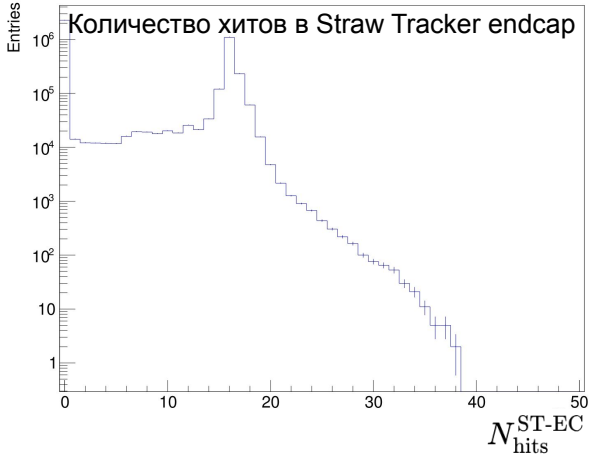


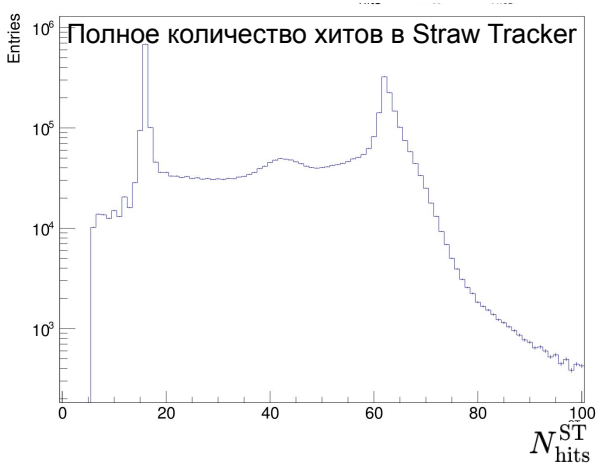
Отобраны треки, у которых есть FirstState, LastState, FinalState
Знак заряда определен по FirstState трека
Значения p_T и η трека взяты из FinalState (т.е. состояния трека в первичной вершине)

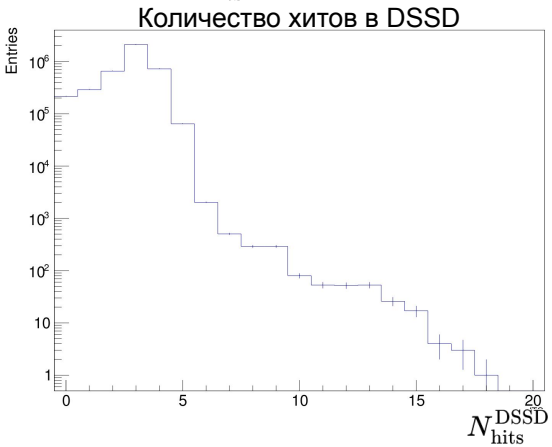
Количество хитов, используемых для фита трека p + p √s = 27 ГэВ, spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()

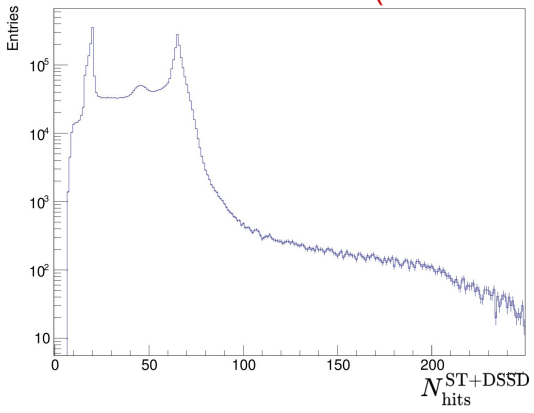






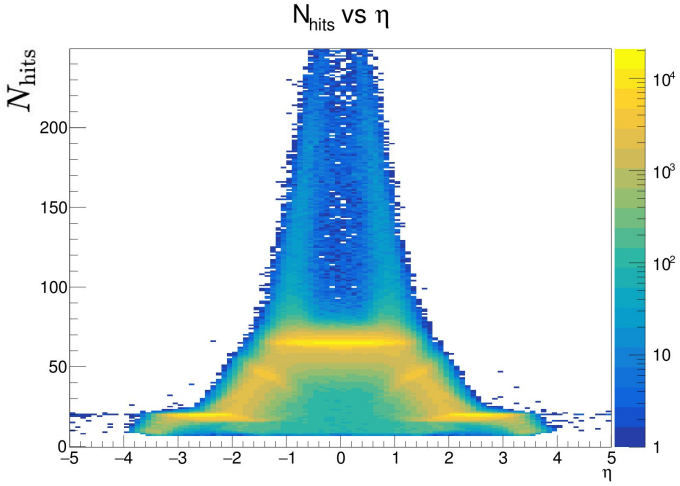


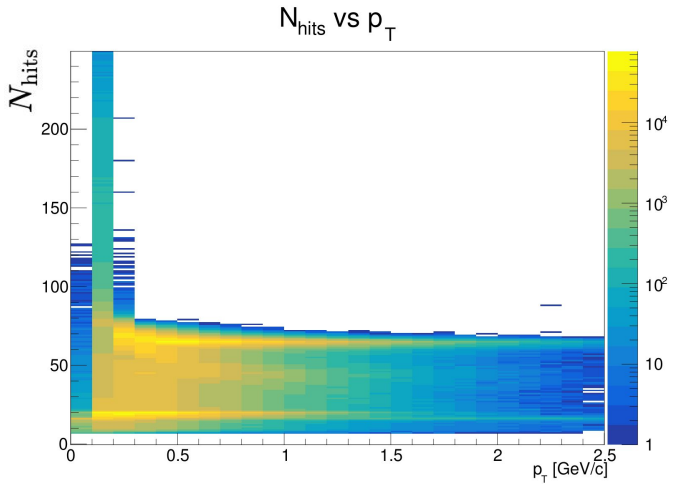
Полное количество хитов (в DSSD и ST)

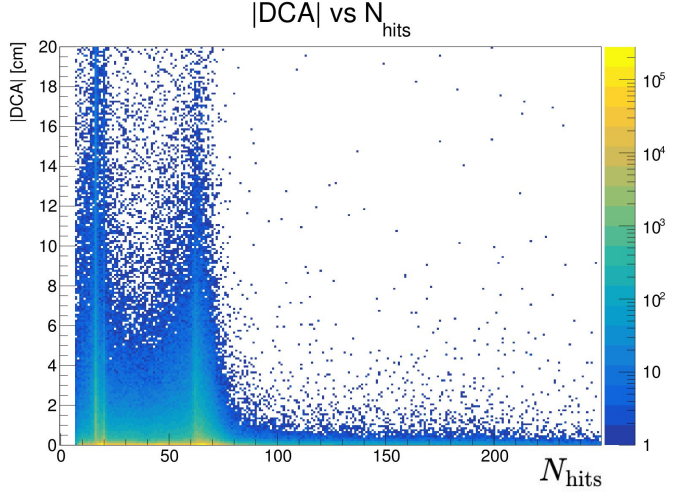


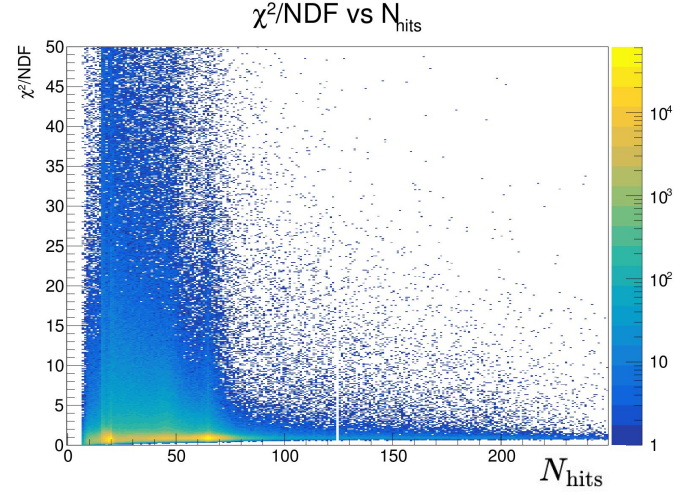
p + p √s = 27 ГэВ, spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()









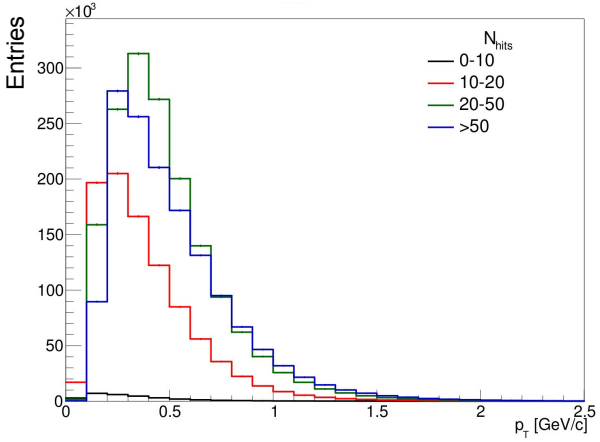
N_{hits} – сумма хитов в ST и DSSD

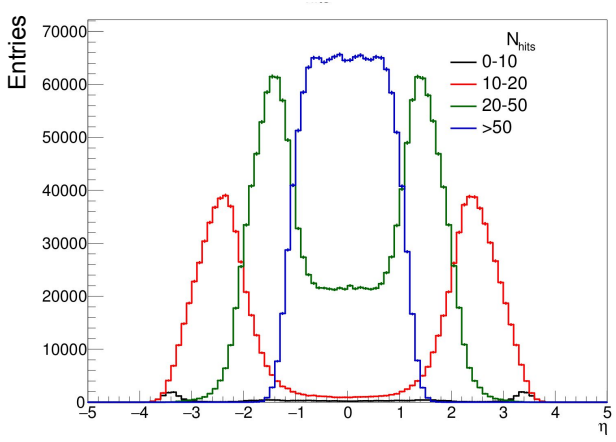
χ^2/ndf – качество фита отдельного трека

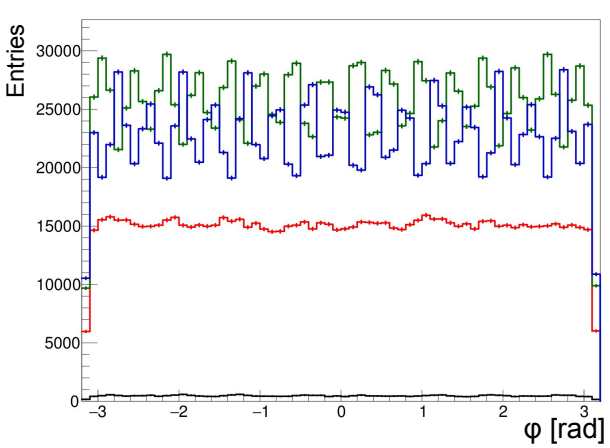
Распределения p_T, η, φ и DCA для разного количества хитов

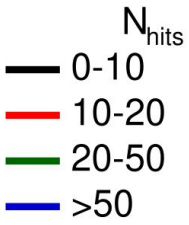
p + p √s = 27 ГэВ, spdroot-4.1.7.6

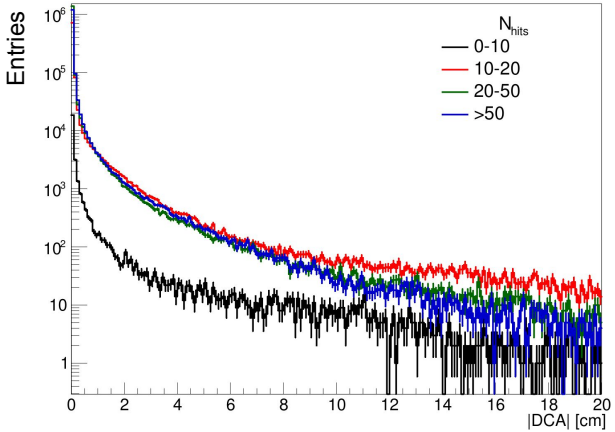
- SpdTrackMC: Final State, isGood()



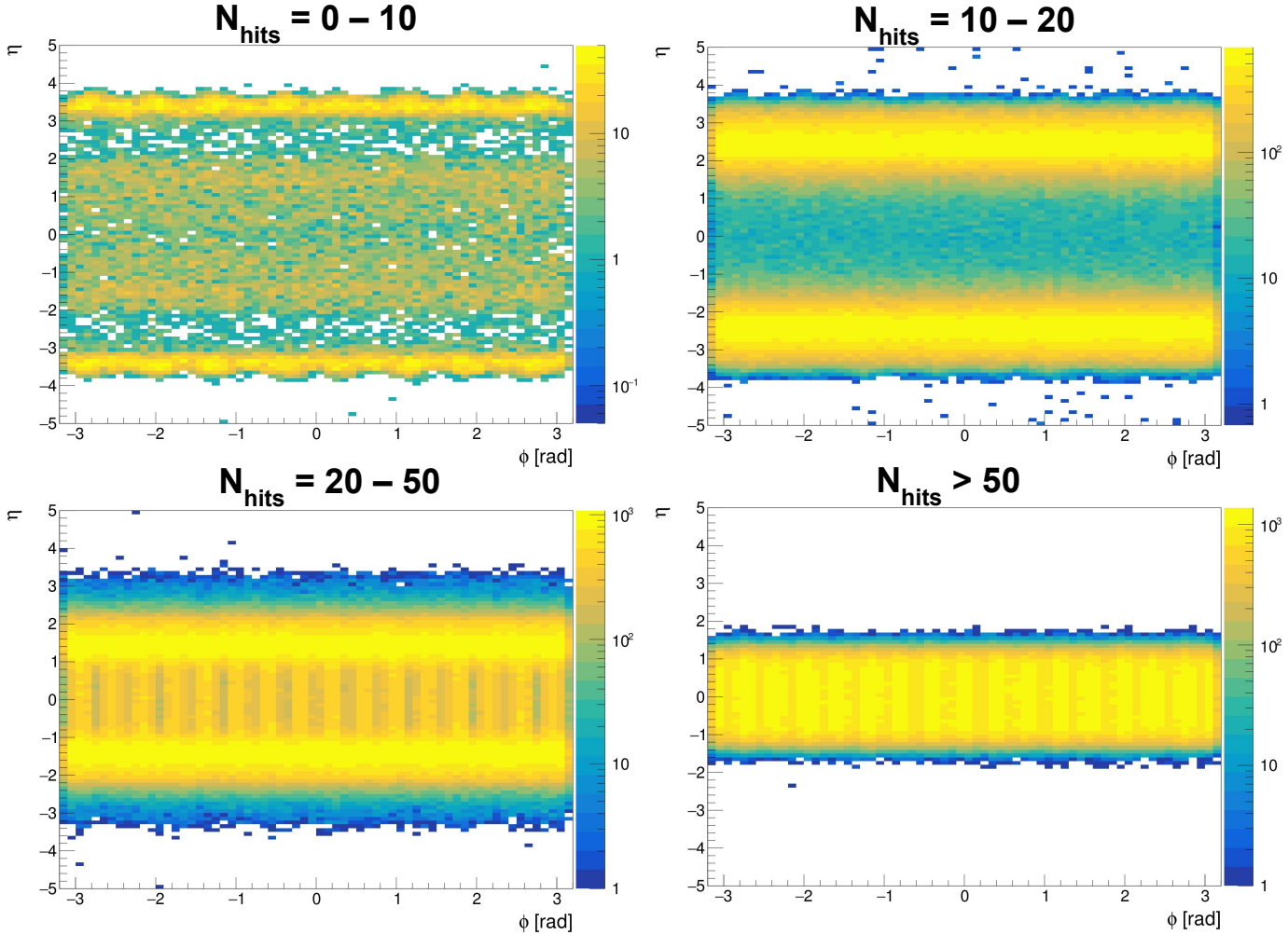








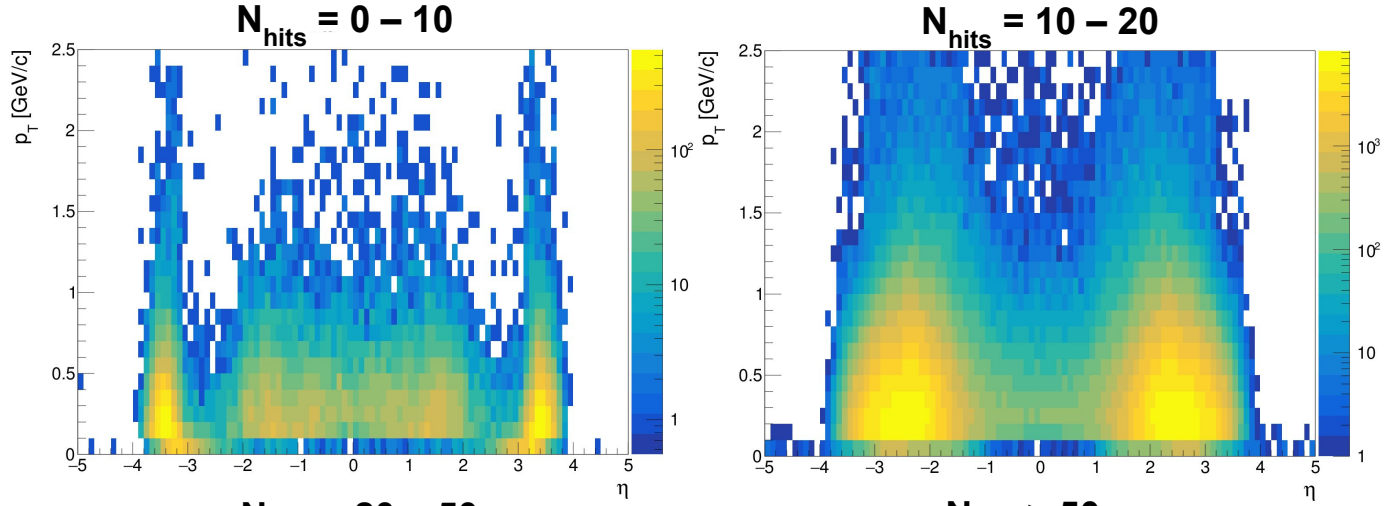
Распределение η vs. φ для разного количества хитов

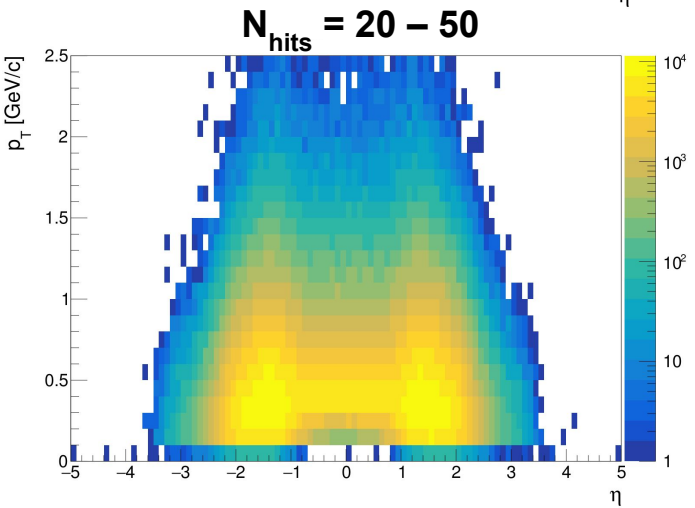


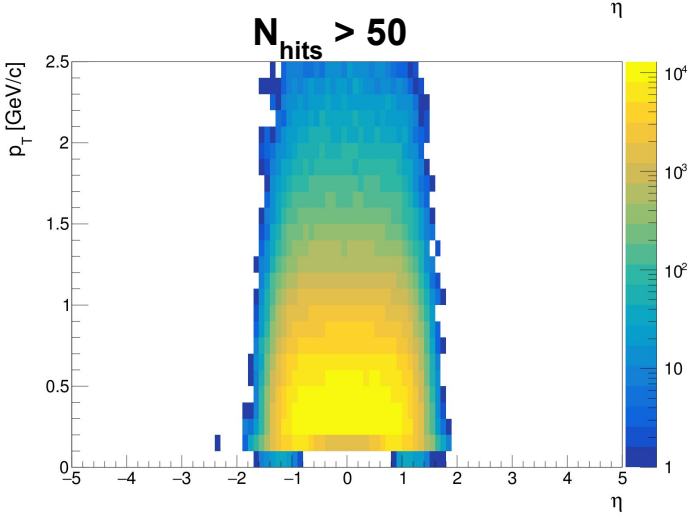
p + p $\sqrt{s}$ = 27 ГэВ,
spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()

Распределение p_T vs. η для разного количества хитов



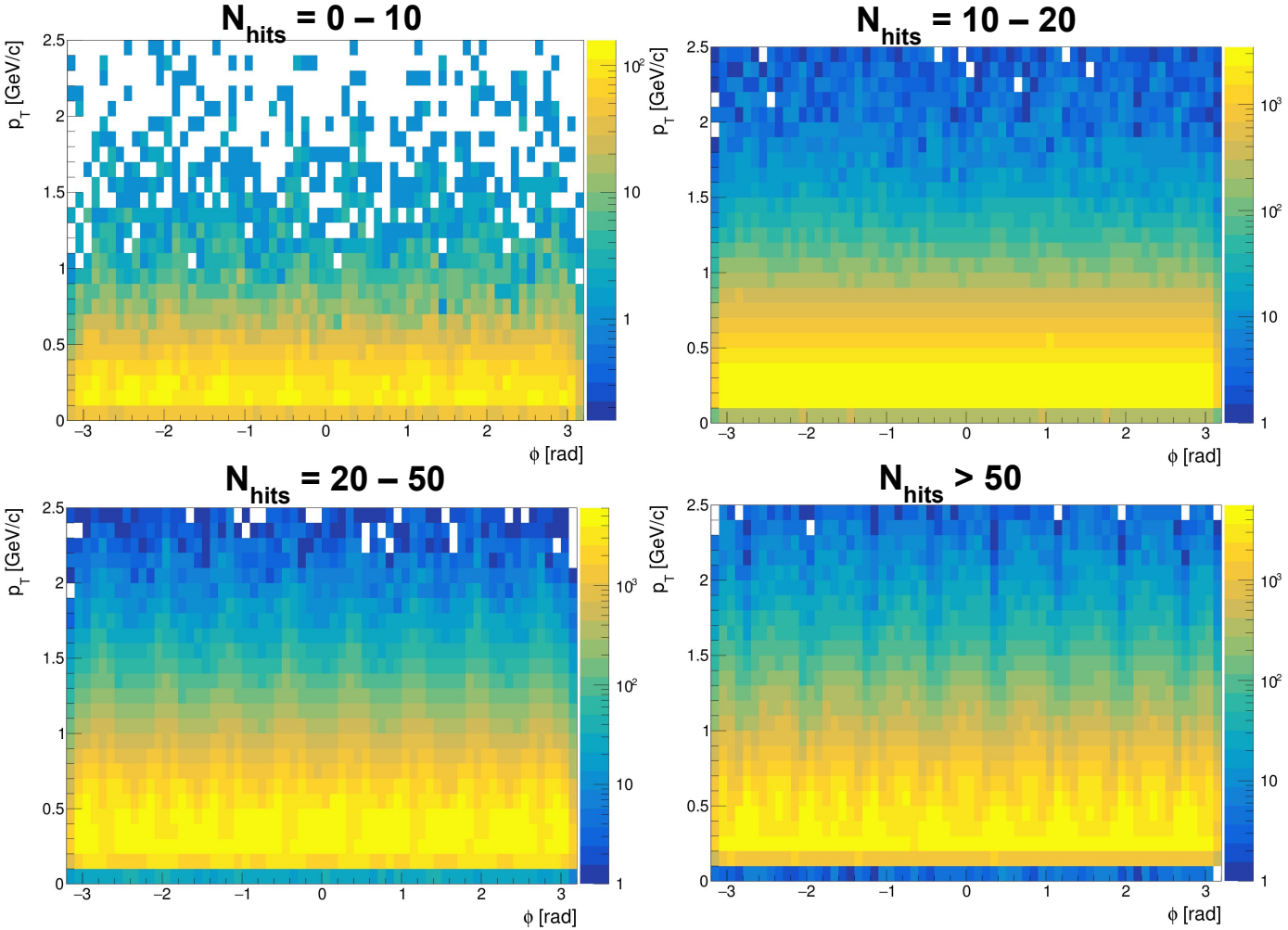




p + p √s = 27 ГэВ,
spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()

Распределение p_T vs. φ для разного количества хитов



p + p $\sqrt{s}$ = 27 ГэВ,
spdroot-4.1.7.6

- SpdTrackMC: Final State, isGood()

Backup

SpdTrackFitPar.h

```
//-----------------------------------------------------------------------------
inline Bool_t SpdTrackFitPar::GetIsFittedOk() const
{
    if (fErrorFlag != 0) return false;
    if (!fFirstState) return false;
    if (!fLastState) return false;
    return true;
}

//-----------------------------------------------------------------------------
inline Bool_t SpdTrackFitPar::GetIsGood() const
{
    if (fErrorFlag != 0) return false;
    if (HasErrorMesg()) return false;
    //if (fNFailedHits > 0) return false;
    if (fConvergencyGF != 1) return false; // 0 (not converged), -1 (partially converged), 1(fully converged)
    return true;
}

//-----------------------------------------------------------------------------
inline Bool_t SpdTrackFitPar::GetIsAcceptable() const
{
    if (fErrorFlag != 0) return false;
    if (HasErrorMesg()) return false;
    //if (fNFailedHits > 0) return false;
    if (fNDF < 3) return false;
    if (GetChi2overNDF() < 2) return true;
    return false;
}
```