Single- versus Double-Precision in Traccc

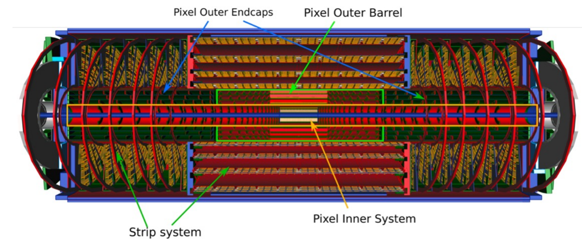
Yuki Asami (Kobe U.)
Shima Shimizu (KEK)
Yuji Yamazaki (Kobe U.)

Introduction 1/2

The new ATLAS inner detector

- The ATLAS inner detector will be upgraded.
  →The number of its readout channels of will significantly increase.

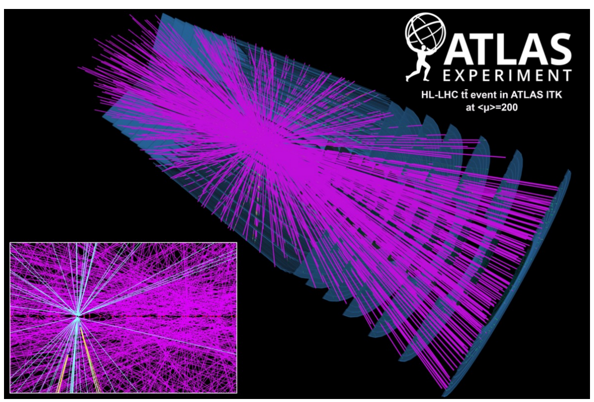
All silicon



The new ATLAS inner detector (ITk: Inner Tracker)

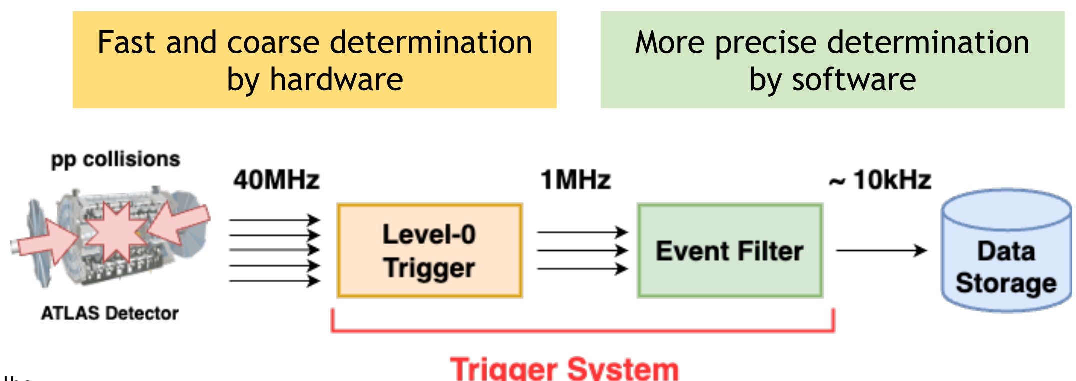
Challenges for the HL-LHC and GPU tracking

- The track reconstruction of the HL-LHC will face huge computation.
  →Several methods are explored for the track reconstruction in Event Filter.
- One of these tracking methods is GPU tracking(Traccc)!



https://atlas.cern/updates/news/scientific-potential-high-luminosity-lhc

200 simultaneous pp collisions per bunch crossing at the HL-LHC!



The ATLAS trigger system for HL-LHC

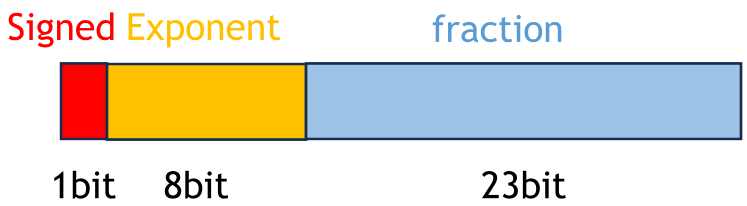
Introduction 2/2

My study so far and this presentation

- I have been working on the performance comparison between CPUs and GPUs(CUDA) so far.
- This time, I will present on the performance comparison in terms of processors(CPU or GPU) and calculation precision(single or double).

Calculation precision

- Single precision 32bit (In this presentation, I will use "float" instead of "single".)

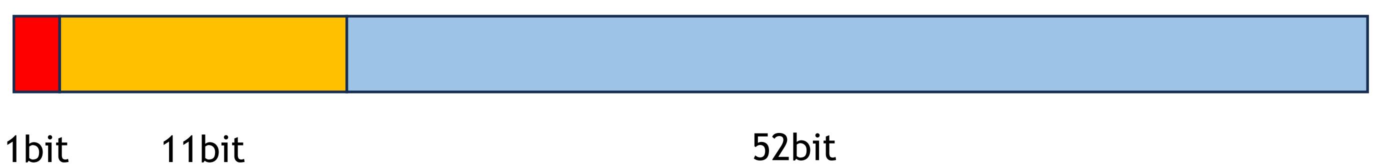


Signed fraction Exponent

$+ 2.99792458 \times 10^{8}$

※It is actually represented in binary.

- Double precision 64bit



I will use notations such as CPU-double, GPU-float etc… in this presentation.

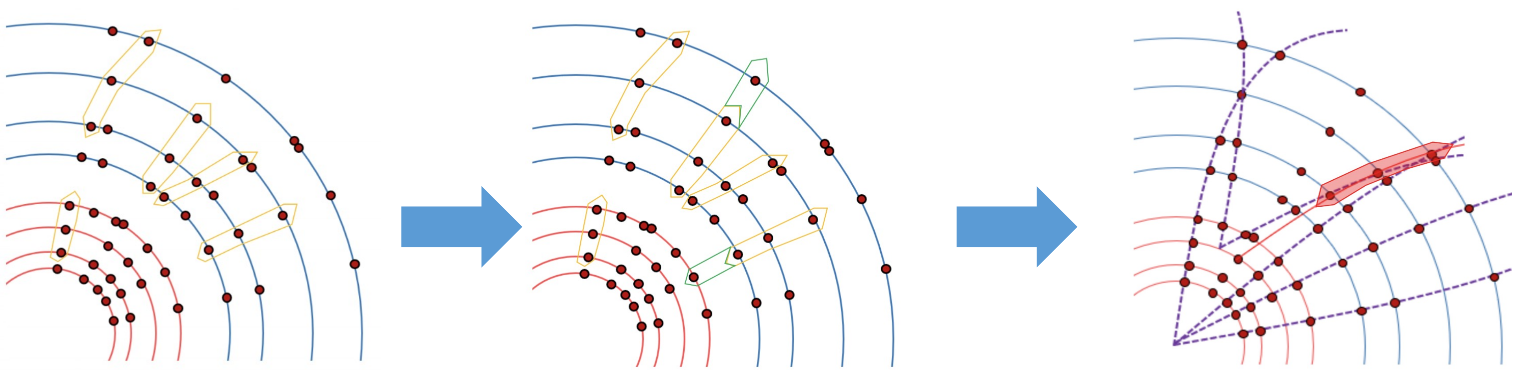
Algorithm for the tracking

Tracking chain

On GPU (not only GPU) | On CPU

Clustering ⇨ **Seeding** ⇨ **Finding** ⇨ **Fitting** ⇨ Ambiguity resolution ⇨ Physical analysis

⟵ **The range of my study** ⟶ (Seeding, Finding, Fitting)



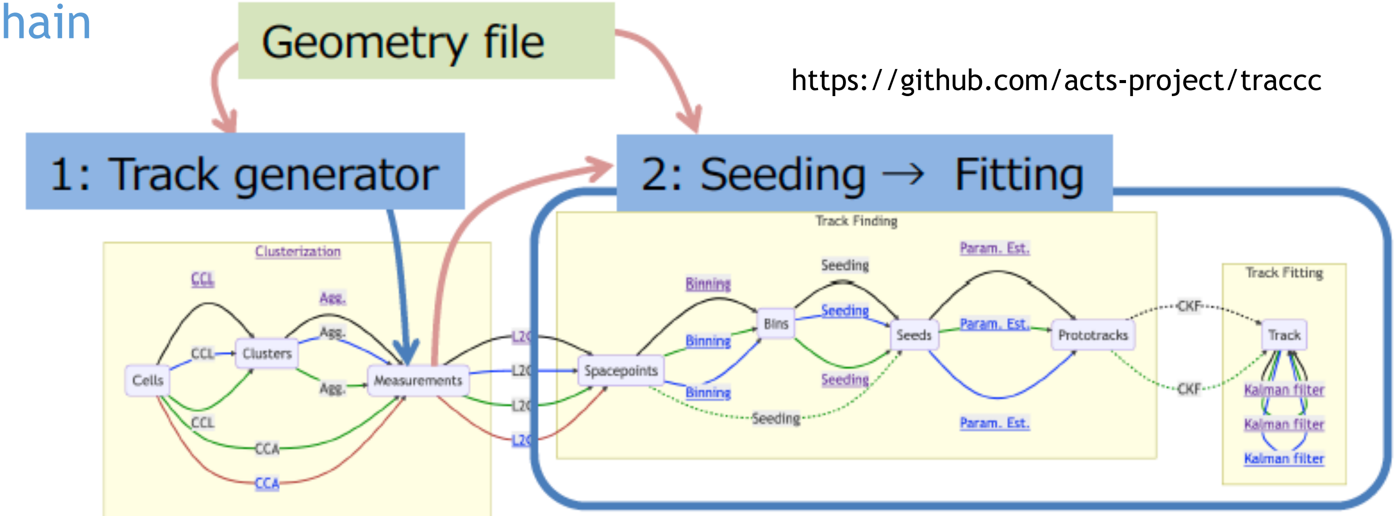
Seeding | Finding | Fitting

Seeding : Making a set of three compatible hits(= a seed)
Finding : Connecting hits compatible with seeds
Fitting : Obtaining track parameters

The setup of the study

Testing chain

https://github.com/acts-project/traccc



I used the track generator and generated the tracks from a point(0, 0 ,0).I let these tracks pass through the geometry(ITk). Then, the measurements(hit points on the detector) are created and set as an input of seeding.

Processors

- GPU: NVIDIA RTX A6000
- CPU: Intel(R) Xeon(R) Gold 5318Y CPU @ 2.10GHz

The version of Traccc v0.15.0

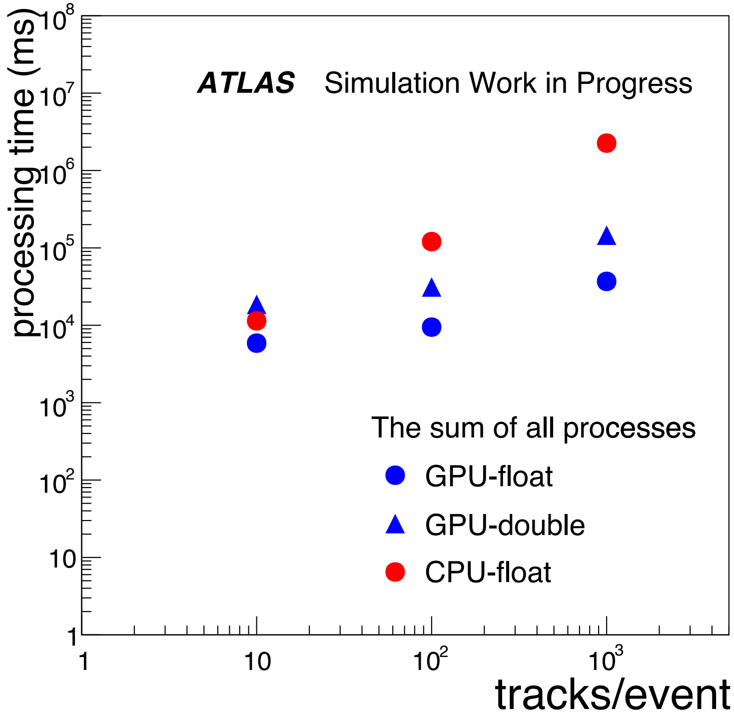
Geometry of the detector ITk (the new ATLAS inner detector)

Generated tracks 100events, 1000tracks/event, 10GeV<p<100GeV, $-4.0<\eta<4.0$

Note: No pileup and no physical particle (massless, q = -1)

Study on the comparison of the processing time

Processing time for various combinations of the processors and the calculation precisions.



GPU is much faster than CPU!

Note : There is no pileup in this study.

The processing time on CPU-double is the same as that on CPU-float.

- GPU is much faster than CPU: about 100x for 1000 tracks
- The more tracks, the more advantageous GPU is.

Question : Is single precision enough?

Numerical precision for seeding 1/2

Matching rates between CPU and GPU

- Matching rate is comparing 3 spacepoints(sps), weight and z-vertex between CPU and GPU.
  - weight : A ranking parameter
  - z-vertex : A track parameter, z0
- CPU and GPU use the same sps, so there is no difference between sps.

Single(float)

```
===>>> Event 10 <<<===
Number of seeds: 21428 (host), 21428 (device)
  Matching rate(s):
    - 3.83144% at 0.01% uncertainty
    - 7.21486% at 0.1% uncertainty
    - 22.8626% at 1% uncertainty
    - 49.9767% at 5% uncertainty
```

Double

```
===>>> Event 10 <<<===
Number of seeds: 21428 (host), 21428 (device)
  Matching rate(s):
    - 100% at 0.01% uncertainty
    - 100% at 0.1% uncertainty
    - 100% at 1% uncertainty
    - 100% at 5% uncertainty
```

Perfectly match

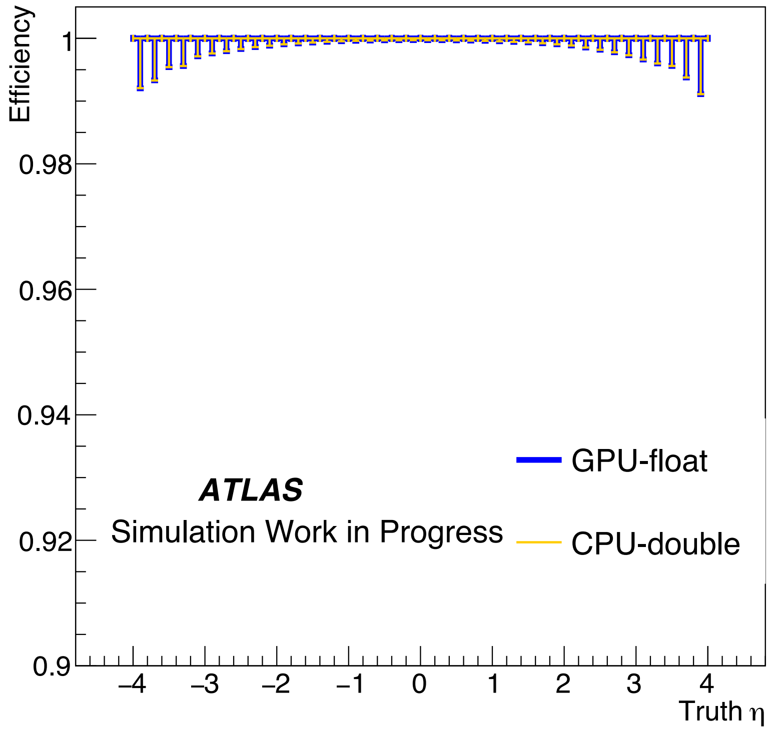
Matching rates on double are 100% at the strongest condition, while matching rates on single are half at maximum.

However, the reason for the low matching rates on single is already understood.

(The difference is due to calculation around zero. →back up, page18~20)

Numerical precision for seeding 2/2

Efficiency (plot only truth $p_T > 1GeV$)



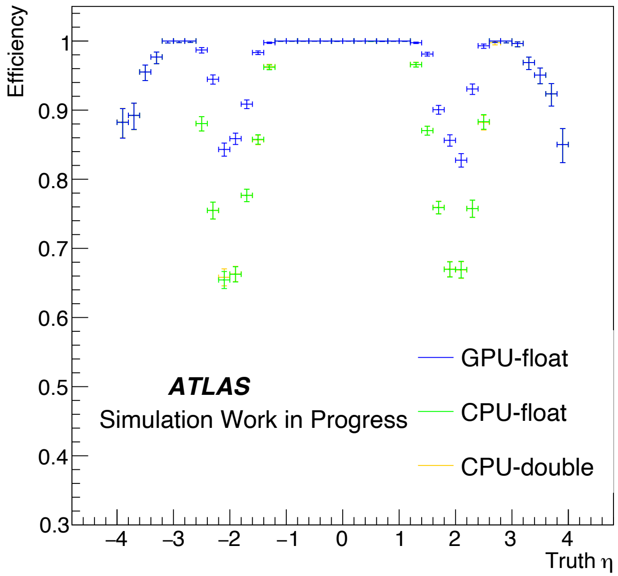
In terms of the efficiency, seeding shows perfect match between CPU-double and GPU-float!

→Single precision(float) seems enough.

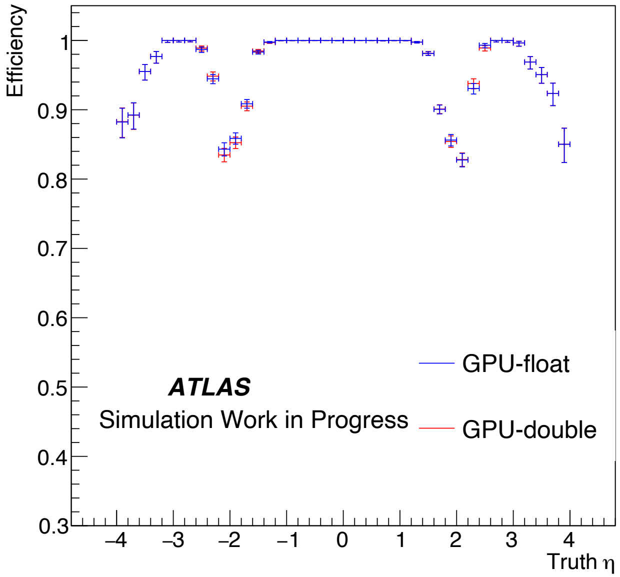
Numerical precision for finding

Efficiency

(plot only truth $p_T > 0.1$GeV)



GPU-float vs CPUs



GPU-float vs GPU-double

- There is a minor difference between GPU and CPU.
  (Need further investigation. Possible reason suggested by experts: non-deterministic order of track candidates on GPU.)
- GPU-float shows almost the same performance as GPU-double.

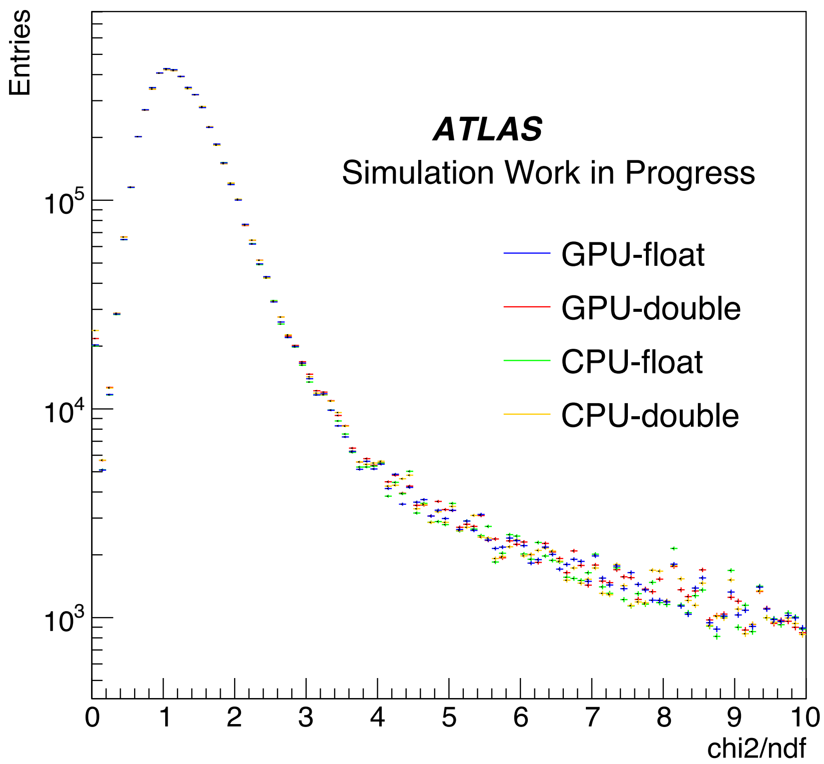
Definition of the efficiency

If there is a reconstructed track which is composed of hits only from a single truth particle, this truth particle is considered as reconstructed and added to the numerator of the efficiency.

Numerical precision for fitting 1/4

From here on, we will look at the fitting parameters.

Chi2/NDF



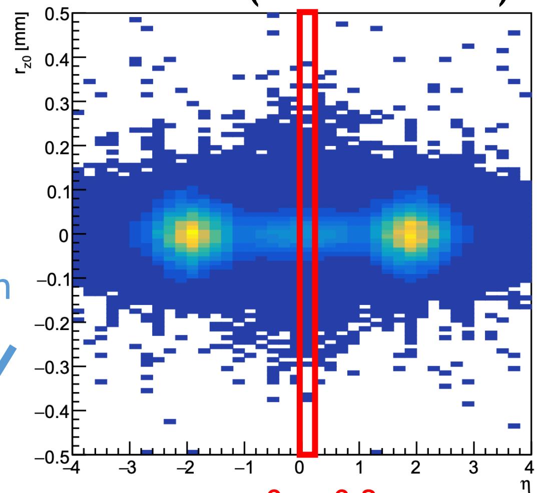
The plots of chi2/NDF show a good match among GPUs and CPUs.

Numerical precision for fitting 2/4

Residual and Resolution

※In this presentation, the residuals are created by Traccc performance codes.
However, the resolutions are created by myself.

Residual(= truth - reco)



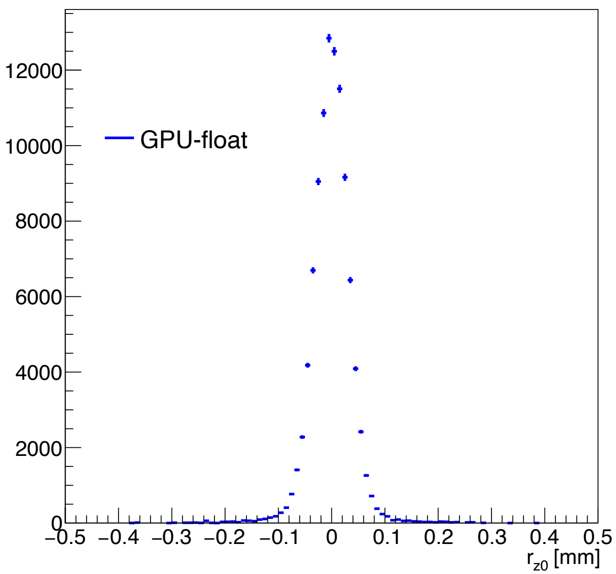
0<η<0.2

Project the bin enclosed by the red frame on the y axis and get the plot at the bottom left.

Imagine the z-axis as the number of events and view it from the side.

Project this distribution on the y axis.

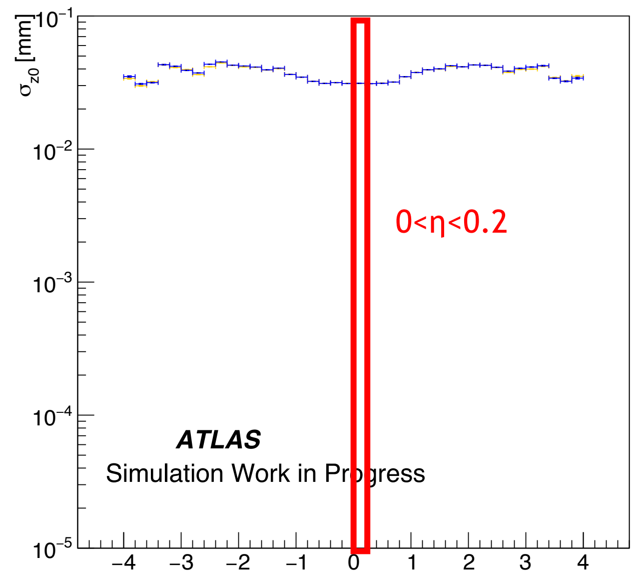
A distribution of residual for a single bin(0<η<0.2)



The sigma obtained from the Gaussian fit of this distribution represents the resolution for 0<η<0.2.

The value of the sigma is assigned to the resolution in the range of 0<η<0.2.

resolution

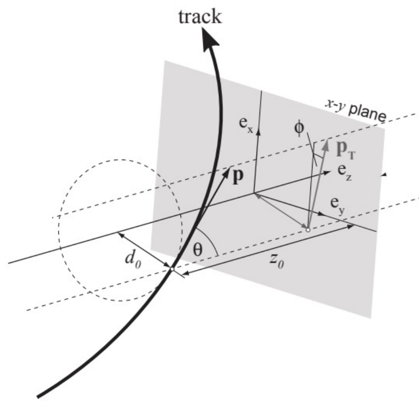


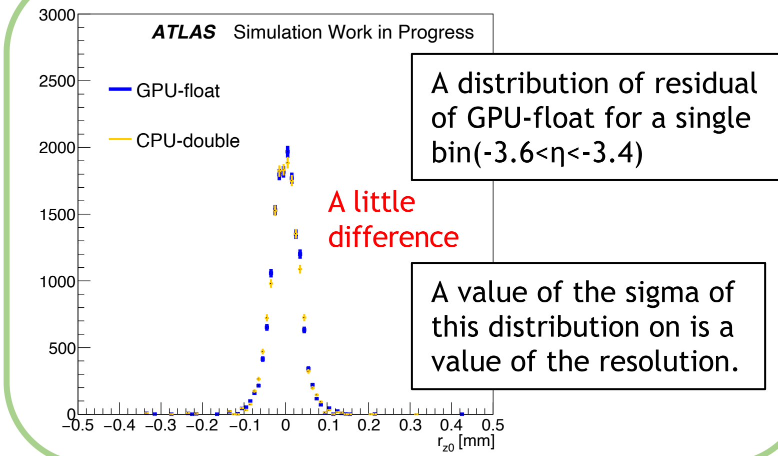
This operation is performed for all η bins to obtain the distribution of the resolution.

Numerical precision for fitting 3/4

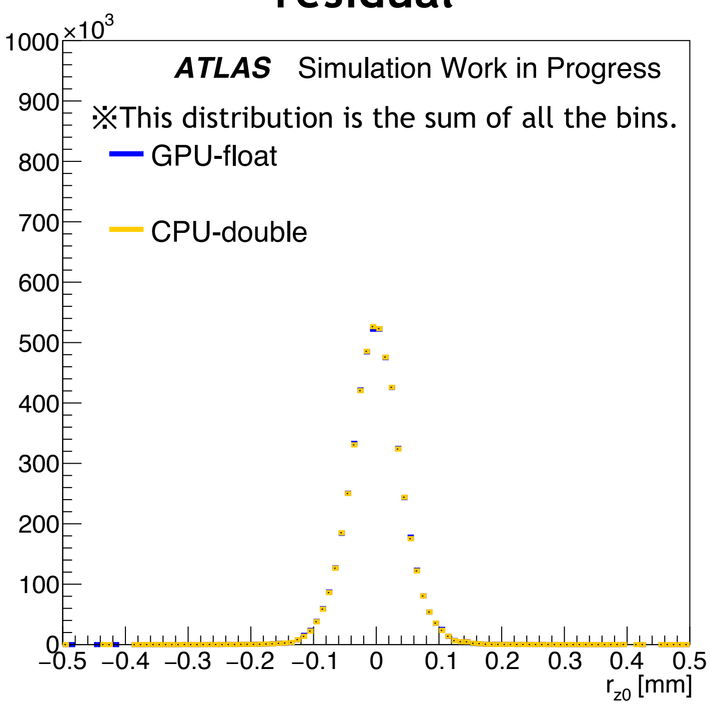
z_0 vs eta

z_0, one of the track parameters represents a distance between the interaction point and the track.



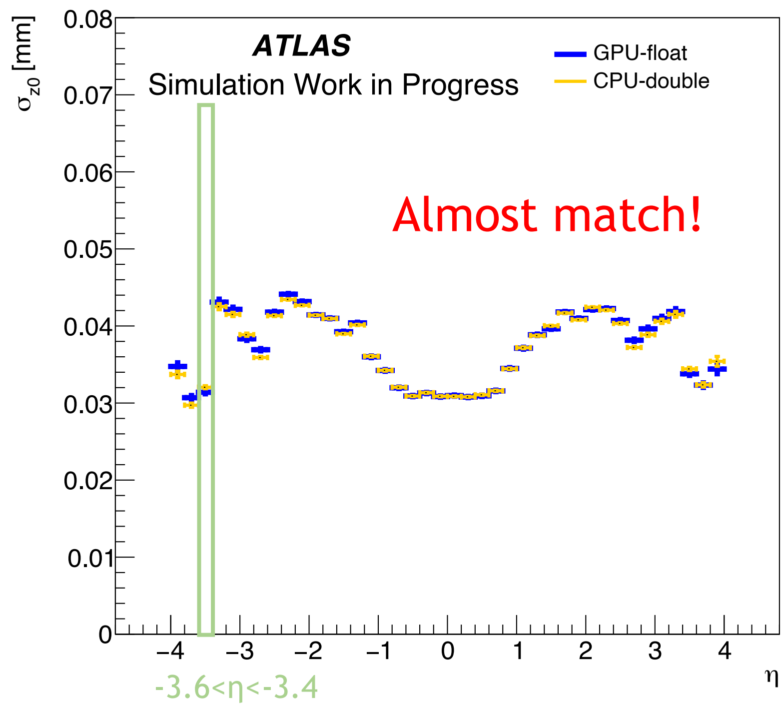


residual



Fit range
|r|<0.1

resolution

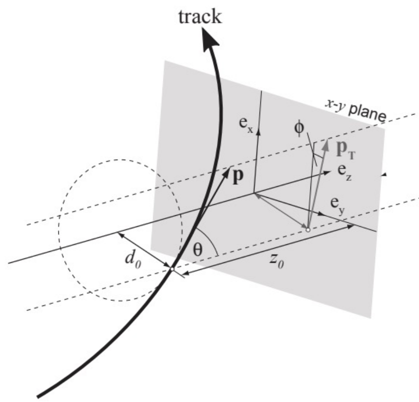


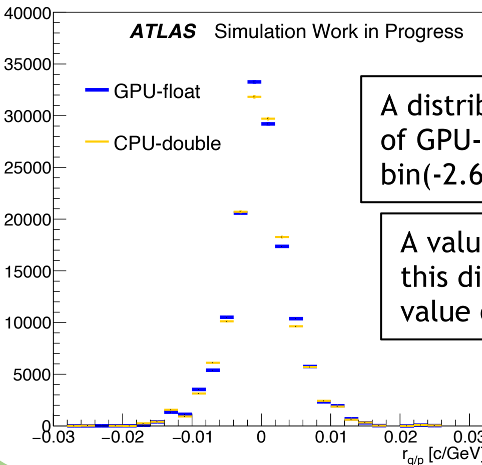
The resolutions of GPU-float seem to be enough in terms of accuracy.

Numerical precision for fitting 4/4

q/p vs eta

q/p, one of the track parameters represents a curvature of a track

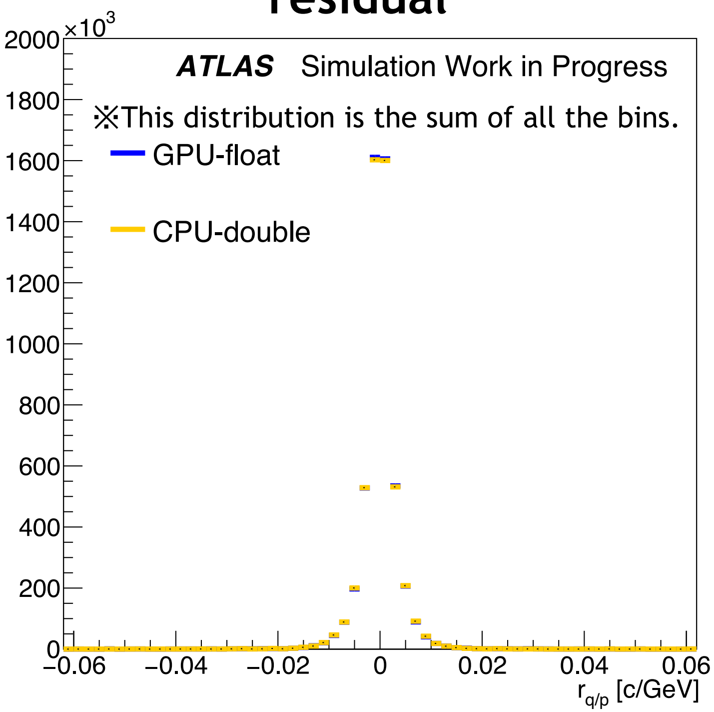




A distribution of residual of GPU-float for a single bin(-2.6<η<-2.4)

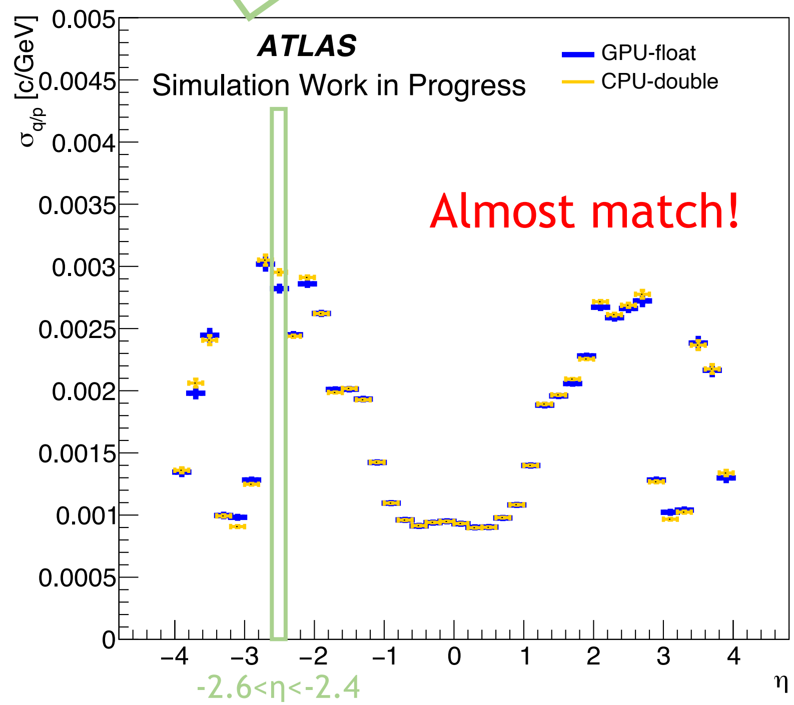
A value of the sigma of this distribution on is a value of the resolution.

residual



Fit range
|r|<0.005

resolution



Almost match!

The resolutions of GPU-float seem to be enough in terms of accuracy.

I checked the other track parameters as well, and there was no issues.(back up)

Conclusion

Topic

- Fast tracking with low power consumption is required at HL-LHC.
  → ACTS on GPU
- This time, I studied performance comparison in terms of processors and calculation precision to look into whether calculation on GPU with single precision is enough.

Result

- GPU is much faster than CPU.
- In terms of calculation precision, single precision(float) seems enough.
- Minor difference of the CKF inefficiency between CPU and GPU will be investigated.

Future prospect

- I plan to work on an optimization of register use in matrix multiplication code.
  (Stephen already gave us instructions. Many thanks!)

Thank you for listening!

Back Up

The design of the processors

NVIDIA RTX 6000

SPECIFICATIONS

GPU memory	**48GB GDDR6**
Memory interface	**384-bit**
Memory bandwidth	**768 GB/s**
Error-correcting code (ECC)	**Yes**
NVIDIA Ampere architecture-based CUDA Cores	**10,752**
NVIDIA third-generation Tensor Cores	**336**
NVIDIA second-generation RT Cores	**84**
Single-precision performance	**38.7 TFLOPS[7]**
RT Core performance	**75.6 TFLOPS[7]**
Tensor performance	**309.7 TFLOPS[8]**
NVIDIA NVLink	**Connects two NVIDIA RTX A6000 GPUs[12]**
NVIDIA NVLink bandwidth	**112.5 GB/s (bidirectional)**

System interface	**PCIe 4.0 x16**
Power consumption	**Total board power: 300 W**
Thermal solution	**Active**
Form factor	**4.4" H x 10.5" L, dual slot, full height**
Display connectors	**4x DisplayPort 1.4a[9]**
Max simultaneous displays	**4x 4096 x 2160 @ 120 Hz, 4x 5120 x 2880 @ 60 Hz, 2x 7680 x 4320 @ 60 Hz**
Power connector	**1x 8-pin CPU**
Encode/decode engines	**1x encode, 2x decode (+AV1 decode)**
VR ready	**Yes**
vGPU software support	**NVIDIA vPC/vApps, NVIDIA RTX Virtual Workstation**
vGPU profiles supported	**See the Virtual GPU Licensing Guide**
Graphics APIs	**DirectX 12 Ultimate, Shader Model 6.6, OpenGL 4.6[10], Vulkan 1.3[10]**
Compute APIs	**CUDA 11.6, DirectCompute, OpenCL 3.0**

NEXT

Intel(R) Xeon(R) Gold 5318Y CPU@2.10GHz

Download Specific

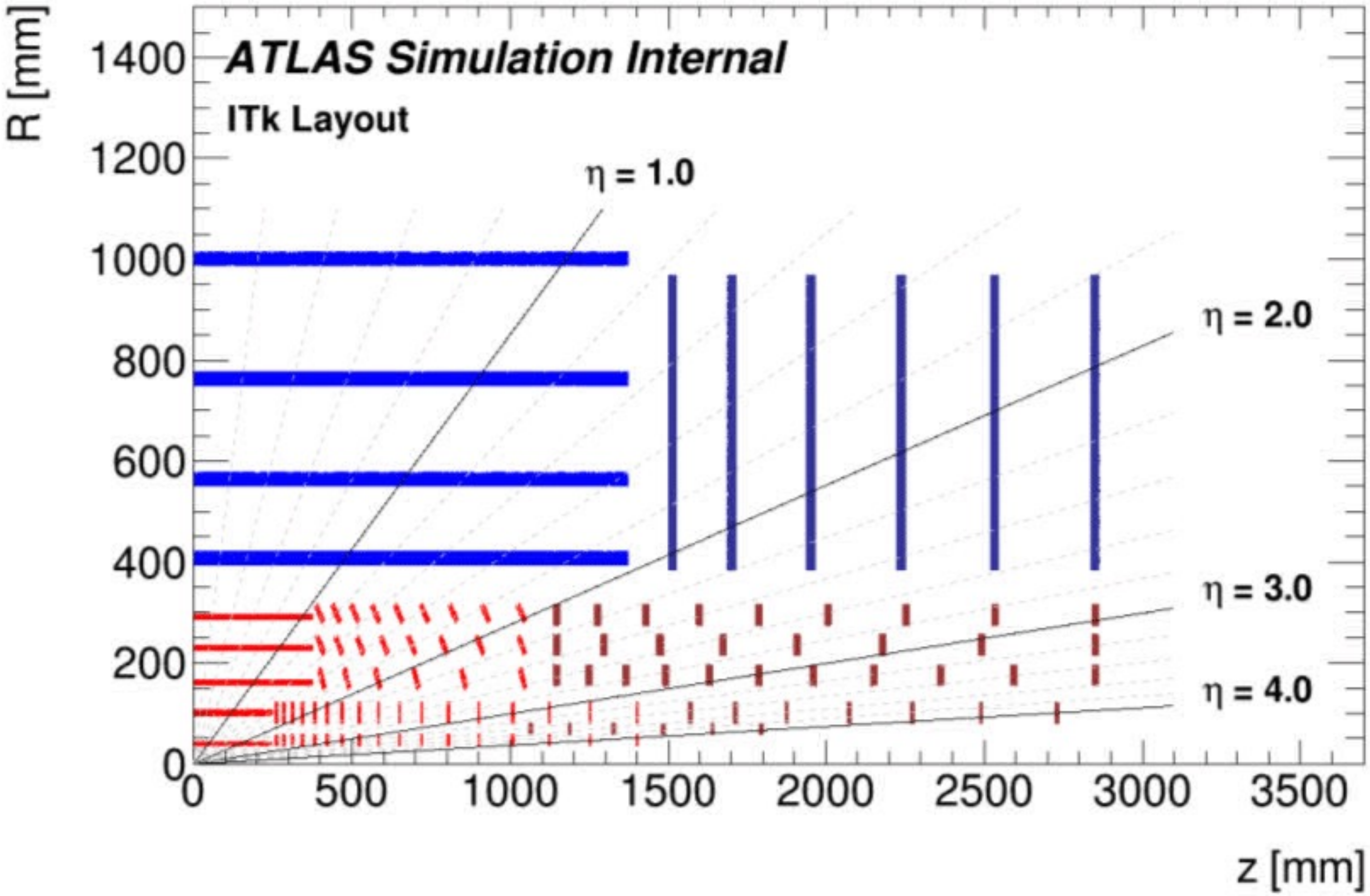
Essentials

Product Collection	3rd Gen Intel® Xeon® Scalable Processors
Code Name	Products formerly Ice Lake
Vertical Segment	Server
Processor Number	5318Y
Lithography	10 nm
Recommended Customer Price	$1483.00

CPU Specifications

Total Cores	24
Total Threads	48
Max Turbo Frequency	3.40 GHz
Intel SpeedStep® Max Frequency	3.40 GHz
Processor Base Frequency	2.10 GHz
Cache	36 MB
Intel® UPI Speed	11.2 GT/s
Max # of UPI Links	3
TDP	165 W

ITk geometry



https://www.researchgate.net/figure/Schematic-layout-of-the-ITk-for-the-HL-LHC-phase-of-ATLAS-Here-only-one-quadrant-and_fig14_333942621

The number of seeds/tracks processed in each process

The number of seeds/tracks processed in each process

	seeding	finding	fitting
GPU-float	2,158,529	5,212,913	5,212,913
GPU-double	2,158,550	5,209,885	5,209,885
CPU-float	2,158,529	5,206,287	5,206,287

The greatest difference among the three processors < 0.13%

Therefore, we can compare the processing time between processors and between calculation precisions.

CPU/GPU Matching rate for seeding 1/3

When I studied performance comparison of CPU and CUDA, I found that the matching rates for seeding are low.

An Important Point in matching rate

The N seeds of CPU and that of CUDA are basically almost equal. In calculation of the matching rates, each value of parameters is compared.

- Matching rate is comparing 3 spacepoints(sps), weight and z-vertex between CPU and CUDA.
  - weight : A ranking parameter
  - z-vertex : A tracking parameter, z0
- CPU and CUDA use the same sps, so there is no difference between sps. I studied weight and z-vertex.

```
===>>> Event 0 <<<===
Number of seeds: 2132 (host), 2132 (device)
  Matching rate(s):
    - 0.140713% at 0.01% uncertainty
    - 2.1576% at 0.1% uncertainty
    - 20.3565% at 1% uncertainty
    - 57.833% at 5% uncertainty
Number of track parameters: 2132 (host), 2132 (device)
  Matching rate(s):
    - 96.8574% at 0.01% uncertainty
    - 99.7655% at 0.1% uncertainty
    - 99.7655% at 1% uncertainty
    - 99.7655% at 5% uncertainty
```

P =50 GeV(!=pT), 0.0<η<2.8,
no c.s.s, 100tracks/event, no pileup

- In traccc it is determined as a match if the following conditions are met.

$$\text{Uncertainty} >= \frac{|weight_{CPU} - weight_{CUDA}|}{\frac{1}{2}\{|\text{weight}_{CPU}| + |weight_{CUDA}|\}} \Rightarrow \textbf{match!}$$

(Uncertainty = 0.01%, 0.1%, 1%, 5%, 10%)

z-vertex is as well

When an absolute value is close to zero, you cannot properly evaluate it by using this conditions. I confirm this in the following slides.

CPU/GPU Matching rate for seeding 2/3

- P =50 GeV(!=pT)
- 0.0<η<2.8
- 1track/event
- --c.s.s=1mm
- gpu2

Show matching rate for weight and that of for z-vertex separately.

==========event 0==========

Matching rate for weight
- 60% (at 0.01% uncertainty)
- 60% (at 0.1% uncertainty)
- 66.6667% (at 1% uncertainty)
- 80% (at 5% uncertainty)
- 100% (at 10% uncertainty)

Matching rate for z-vertex
- 0% (at 0.01% uncertainty)
- 0% (at 0.1% uncertainty)
- 0% (at 1% uncertainty)
- 53.3333% (at 5% uncertainty)
- 80% (at 10% uncertainty)

Matching rate for seeds
- 0% (at 0.01% uncertainty)
- 0% (at 0.1% uncertainty)
- 0% (at 1% uncertainty)
- 40% (at 5% uncertainty)
- 80% (at 10% uncertainty)

Matching rate for z vertex is very low!

The difference of z-vertex appears large because the absolute values of z-vertex close to zero.

I confirm it with tracks generated at a large absolute z-vertex value. (Next slide)

Each value is written down

(spB, spM, spT)	weight(cpu)	weight(cuda)	z_vertex(cpu)	z_vertex(cuda)	difference(weight)	difference(z-vtx)
(0, 1, 3)	200	200	-3.05E-05	-3.09E-05	0	1.19E-02
(0, 1, 5)	200	200	-3.05E-05	-3.09E-05	0	1.19E-02
(0, 1, 4)	200	200	-3.05E-05	-3.09E-05	0	1.19E-02
(0, 1, 2)	199.999	199.999	-3.05E-05	-3.09E-05	0	1.19E-02
(0, 2, 4)	99.9997	99.9994	-3.05E-05	-2.71E-05	3.00001E-06	1.18E-01
(0, 2, 3)	99.9995	99.9998	-3.05E-05	-2.71E-05	3.00001E-06	1.18E-01
(0, 2, 5)	99.9994	99.9991	-3.05E-05	-2.71E-05	3.00002E-06	1.18E-01
(1, 2, 5)	99.9985	99.9987	-6.10E-05	-5.61E-05	2.00003E-06	8.34E-02
(1, 2, 4)	99.9968	99.997	-6.10E-05	-5.61E-05	2.00006E-06	8.34E-02
(0, 3, 5)	-0.00366969	-0.00360138	-4.58E-05	-4.29E-05	0.018789532	6.46E-02
(1, 3, 5)	-0.00812612	-0.00839968	-6.10E-05	-6.30E-05	0.033107021	3.15E-02
(2, 3, 5)	-0.028265	-0.0283532	-9.92E-05	-9.64E-05	0.003115606	2.80E-02
(0, 4, 5)	-0.00125926	-0.00135518	-2.29E-05	2.51E-05	0.07337709	2.00E+00
(1, 4, 5)	-0.00186586	-0.00171867	-5.34E-05	-5.28E-05	0.082125132	1.12E-02
(2, 4, 5)	-0.00866043	-0.00784186	-7.63E-05	-7.36E-05	0.099206837	3.58E-02

CPU/GPU Matching rate for seeding 3/3

- P(!=pT)=50 GeV
- 100tracks/event
- --c.s.s=1mm
- $0.0<\eta<2.8$
- no pileup
- gpu2

To increase the absolute value of z-vertex,
I generated tracks at (x, y, z)=(0, 0, 10).

Vertex(0, 0, 0) (mm)

==========event 0==========

Matching rate for weight
- 75.8443% (at 0.01% uncertainty)
- 76.2195% (at 0.1% uncertainty)
- 80.394% (at 1% uncertainty)
- 89.3058% (at 5% uncertainty)
- 92.7298% (at 10% uncertainty)

Matching rate for z vertex
- 0.140713% (at 0.01% uncertainty)
- 3.2364% (at 0.1% uncertainty)
- 27.3921% (at 1% uncertainty)
- 64.1182% (at 5% uncertainty)
- 76.97% (at 10% uncertainty)

Matching rate for seeds
- 0.140713% (at 0.01% uncertainty)
- 2.1576% (at 0.1% uncertainty)
- 20.3565% (at 1% uncertainty)
- 57.833% (at 5% uncertainty)
- 72.7486% (at 10% uncertainty)

Matching rates increase

Matching rates increase

Vertex(0, 0, 10) (mm)

==========event 0==========

Matching rate for weight
- 76.8714% (at 0.01% uncertainty)
- 77.5432% (at 0.1% uncertainty)
- 82.0058% (at 1% uncertainty)
- 89.9232% (at 5% uncertainty)
- 93.1862% (at 10% uncertainty)

Matching rate for z vertex
- 98.6084% (at 0.01% uncertainty)
- 98.6084% (at 0.1% uncertainty)
- 98.6084% (at 1% uncertainty)
- 98.6084% (at 5% uncertainty)
- 98.6084% (at 10% uncertainty)

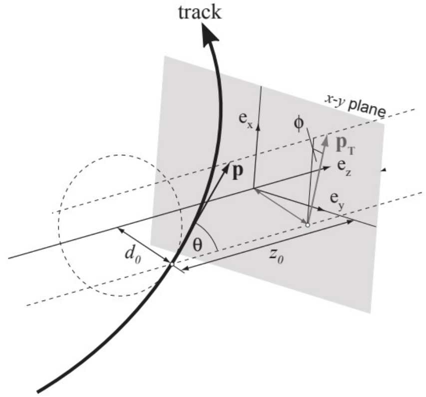
Matching rate for seeds
- 76.8714% (at 0.01% uncertainty)
- 77.5432% (at 0.1% uncertainty)
- 82.0058% (at 1% uncertainty)
- 89.9232% (at 5% uncertainty)
- 93.1862% (at 10% uncertainty)

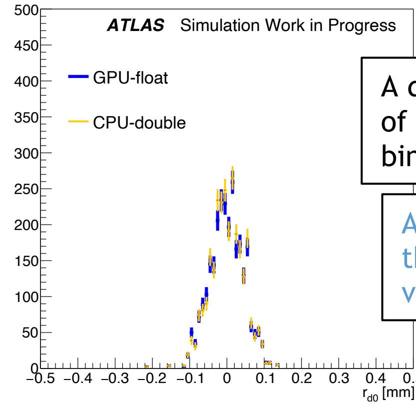
The low matching rates for seeding are the artificial effect of the definition of the matching rate. In fact, the values of CPU and CUDA are matched!

Numerical precision for fitting (back up) 1/3

d_0 vs eta

d_0, one of the track parameters represents a distance between a beam axis and a track.

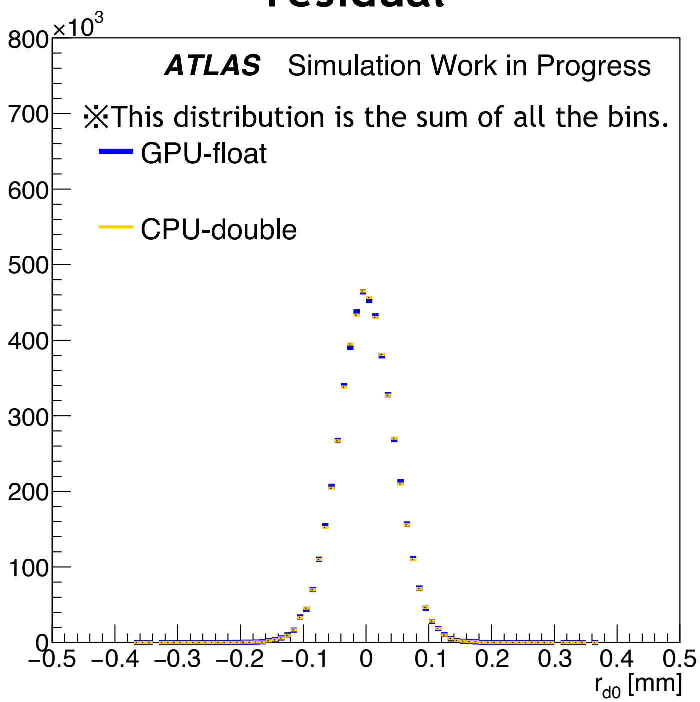




A distribution of residual of GPU-float for a single bin(3.8<η<4.0)

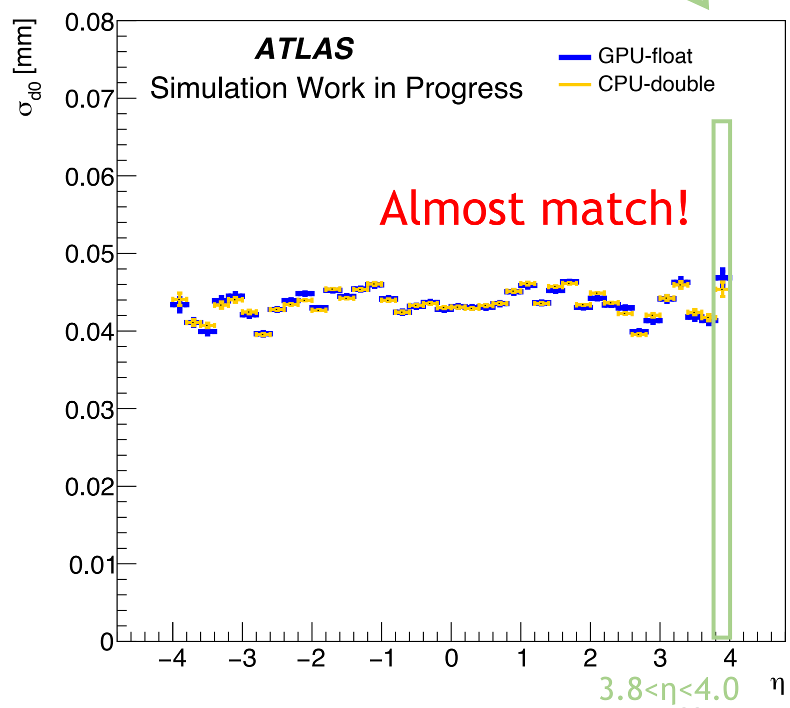
A value of the sigma of this distribution on is a value of the resolution.

residual



Fit range
|r|<0.1

resolution

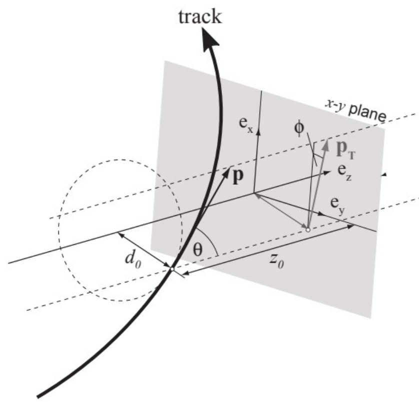


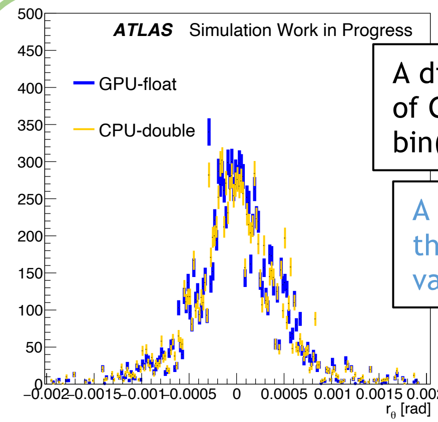
The resolutions of GPU-float seem to be enough in terms of accuracy.

Numerical precision for fitting (back up) 2/3

theta vs eta

theta, one of the track parameters represents an angle from the reference plane.

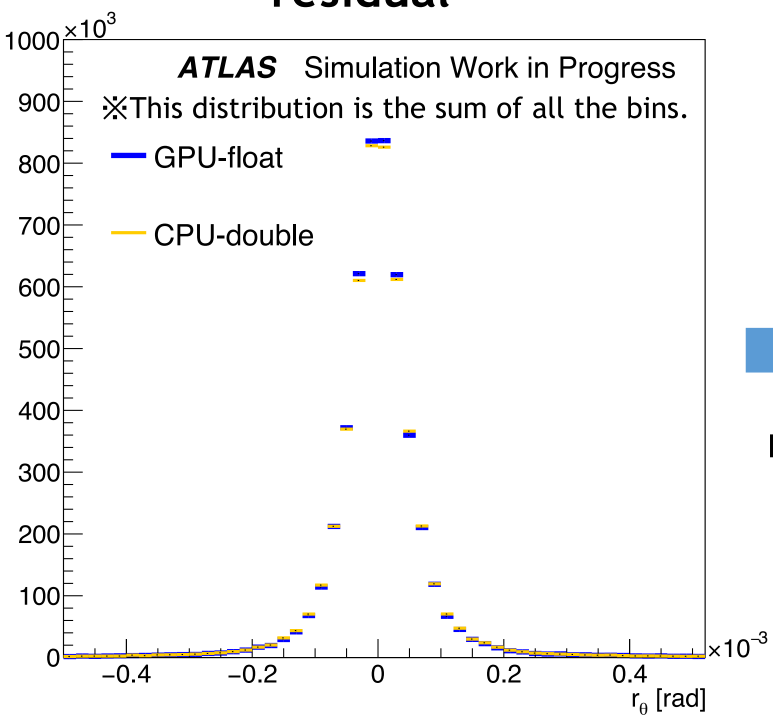




A distribution of residual of GPU-float for a single bin(3.2<η<3.4)

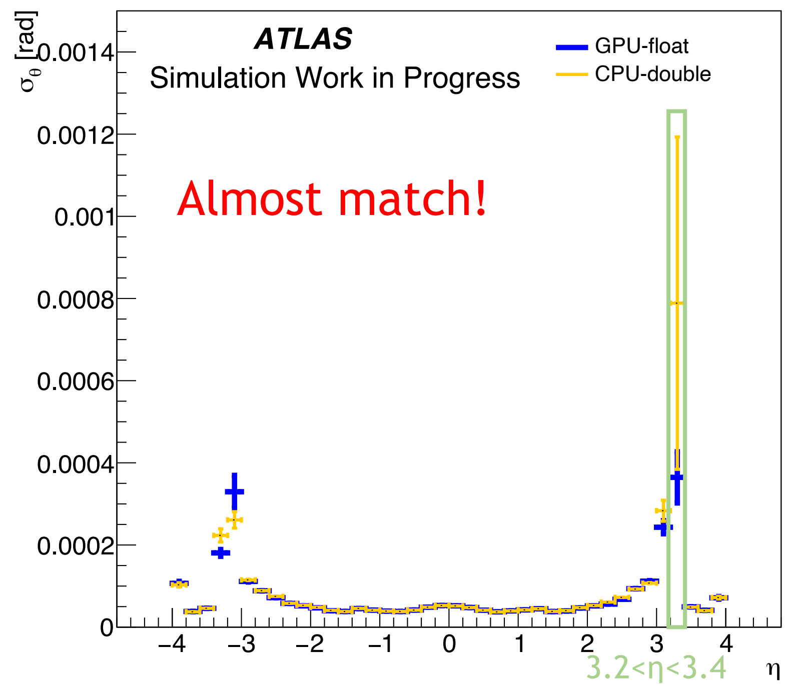
A value of the sigma of this distribution on is a value of the resolution.

residual



Fit range
|r|<0.1

resolution

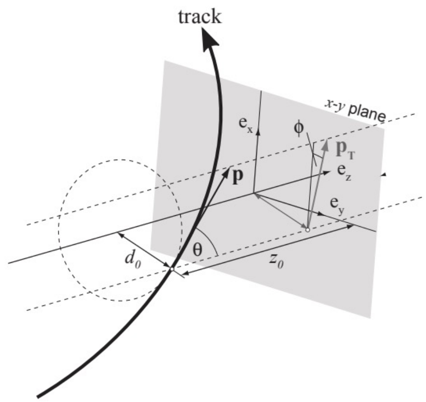


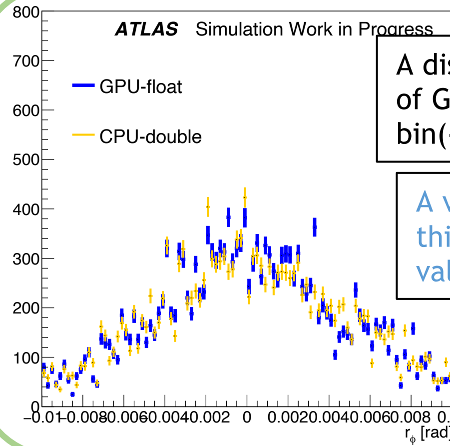
The resolutions of GPU-float seem to be enough in terms of accuracy.

Numerical precision for fitting (back up) 3/3

phi vs eta

phi, one of the track parameters represents azimuthal angle.

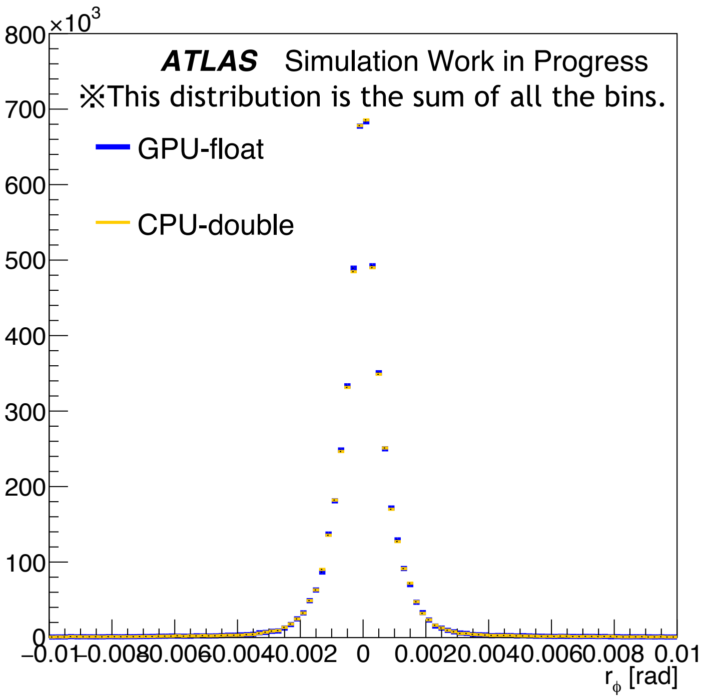




A distribution of residual of GPU-float for a single bin(-3.2<η<-3.0)

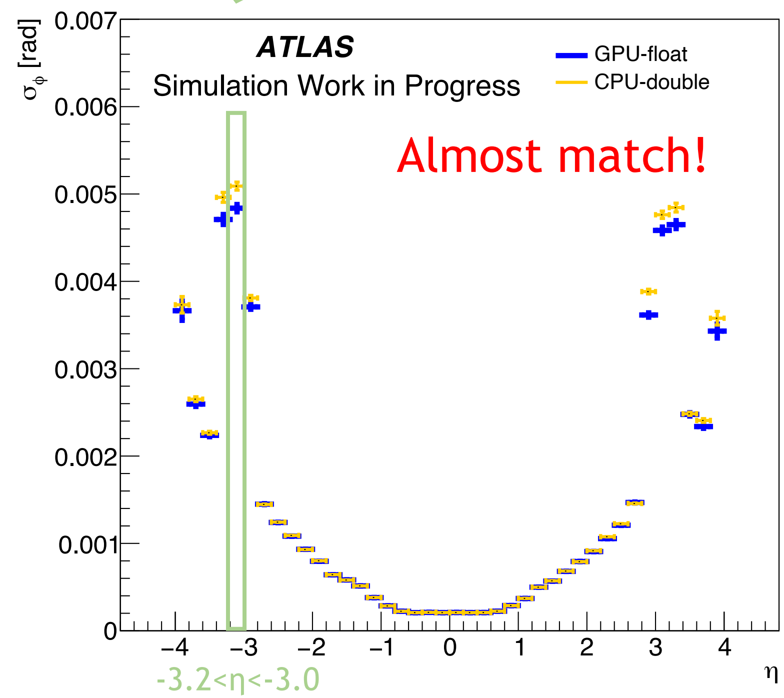
A value of the sigma of this distribution on is a value of the resolution.

residual



Fit range
|r|<0.1

resolution

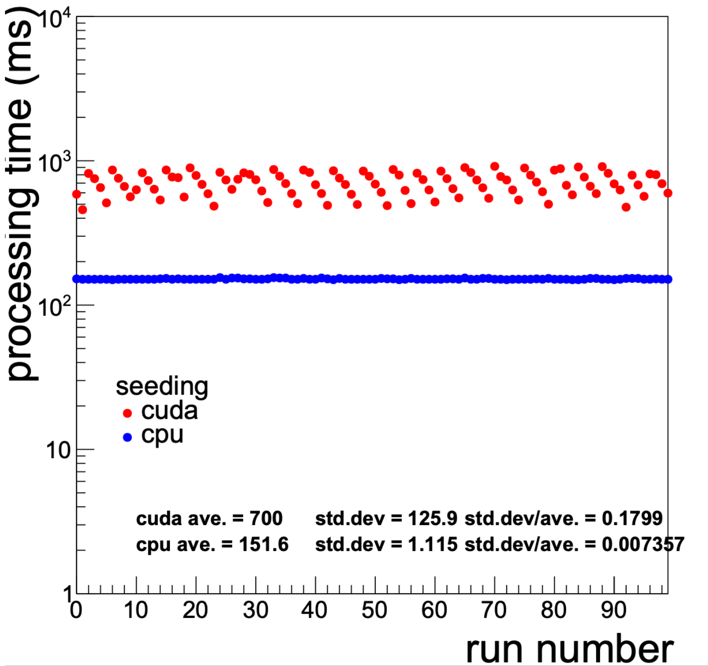


The resolutions of GPU-float seem to be enough in terms of accuracy.

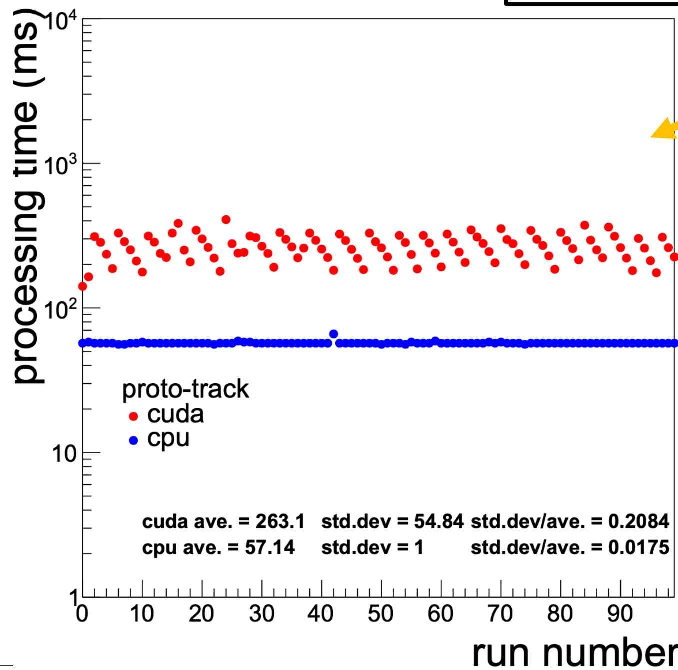
Distributions of processing time with double precision 1/2

10~100GeV, $-4.0<\eta<4.0$, step size = default, max step counts = default, 100tracks/event, 100events/run

If using shell script, blocks of 4 runs appear.



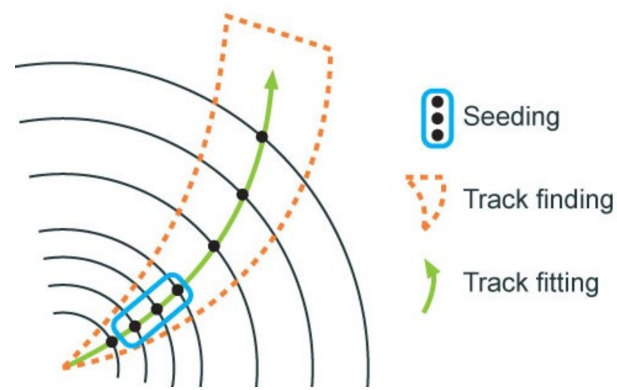
seeding



proto-track

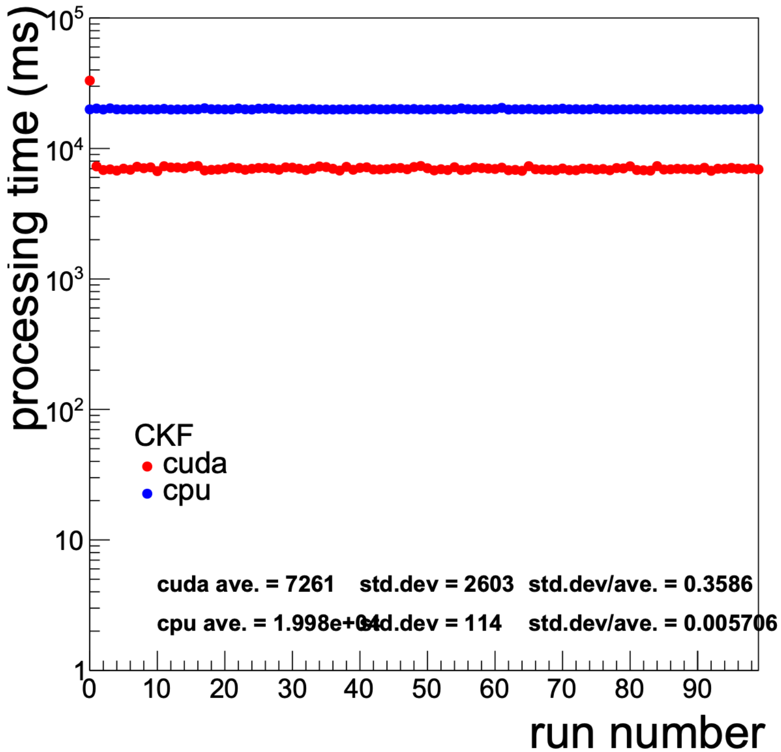
The distribution of double precision is the same as that of float.

In both seeding and proto-track, the deviations are about 20% on CUDA, about 1% on CPU.

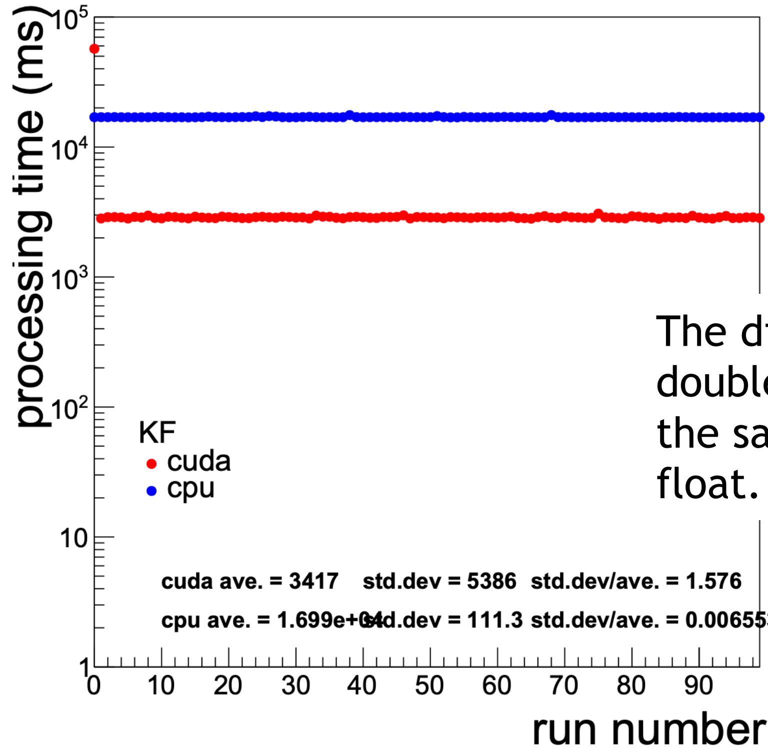


Distributions of processing time with double precision 2/2

10~100GeV, -4.0<η<4.0, step size = default, max step counts = default, 100tracks/event, 100events/run



CKF

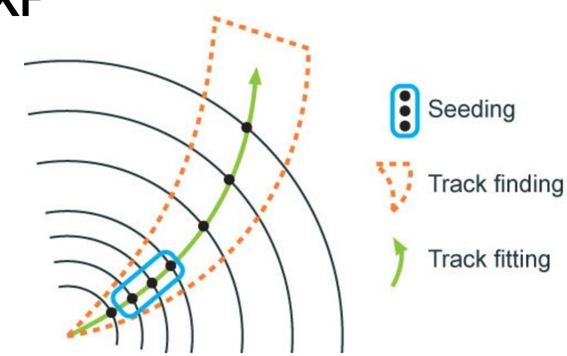


KF

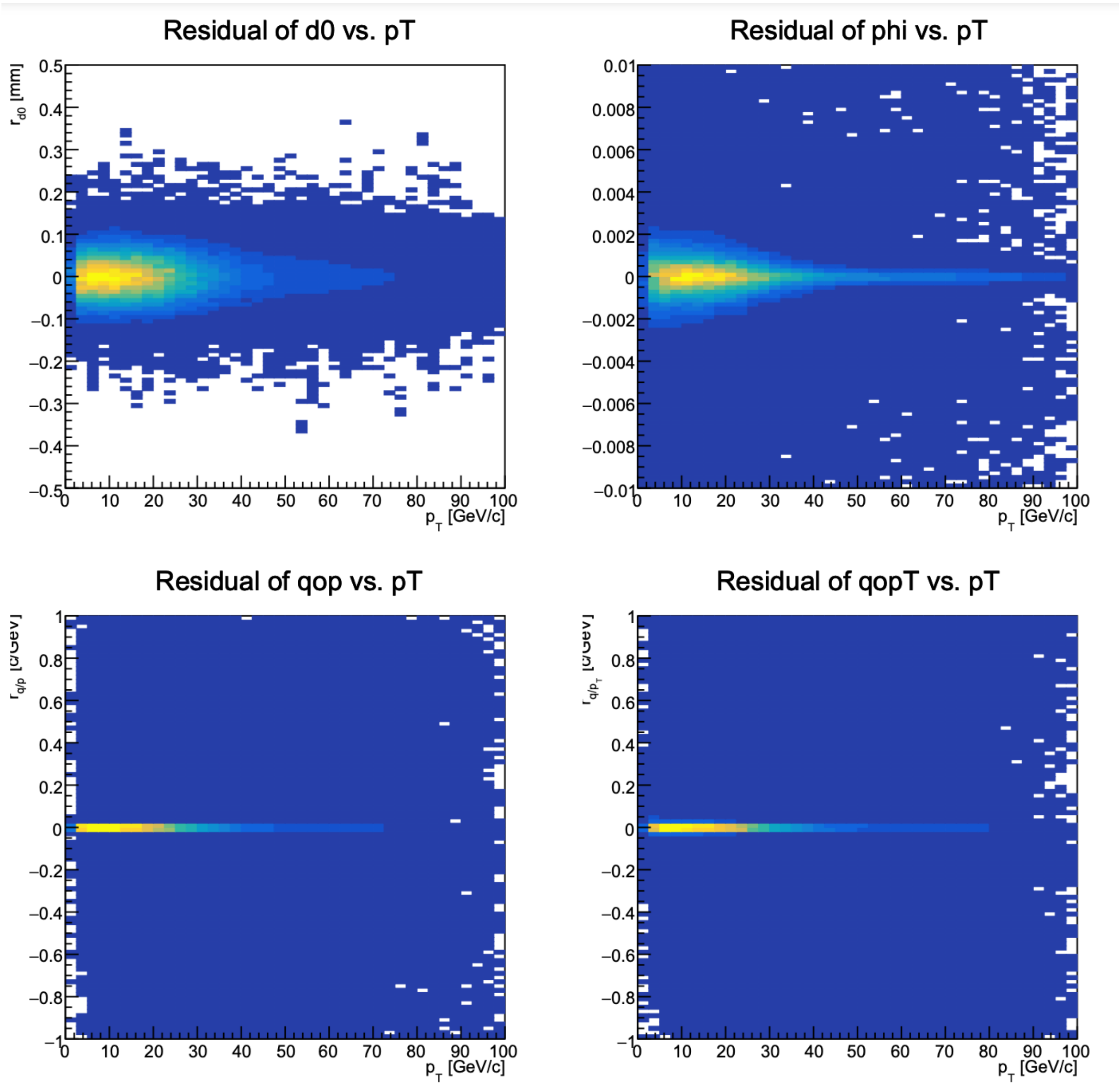
The distribution of double precision is the same as that of float.

In CKF, the deviation is about 3.6% on CUDA, about 0.6% on CPU.

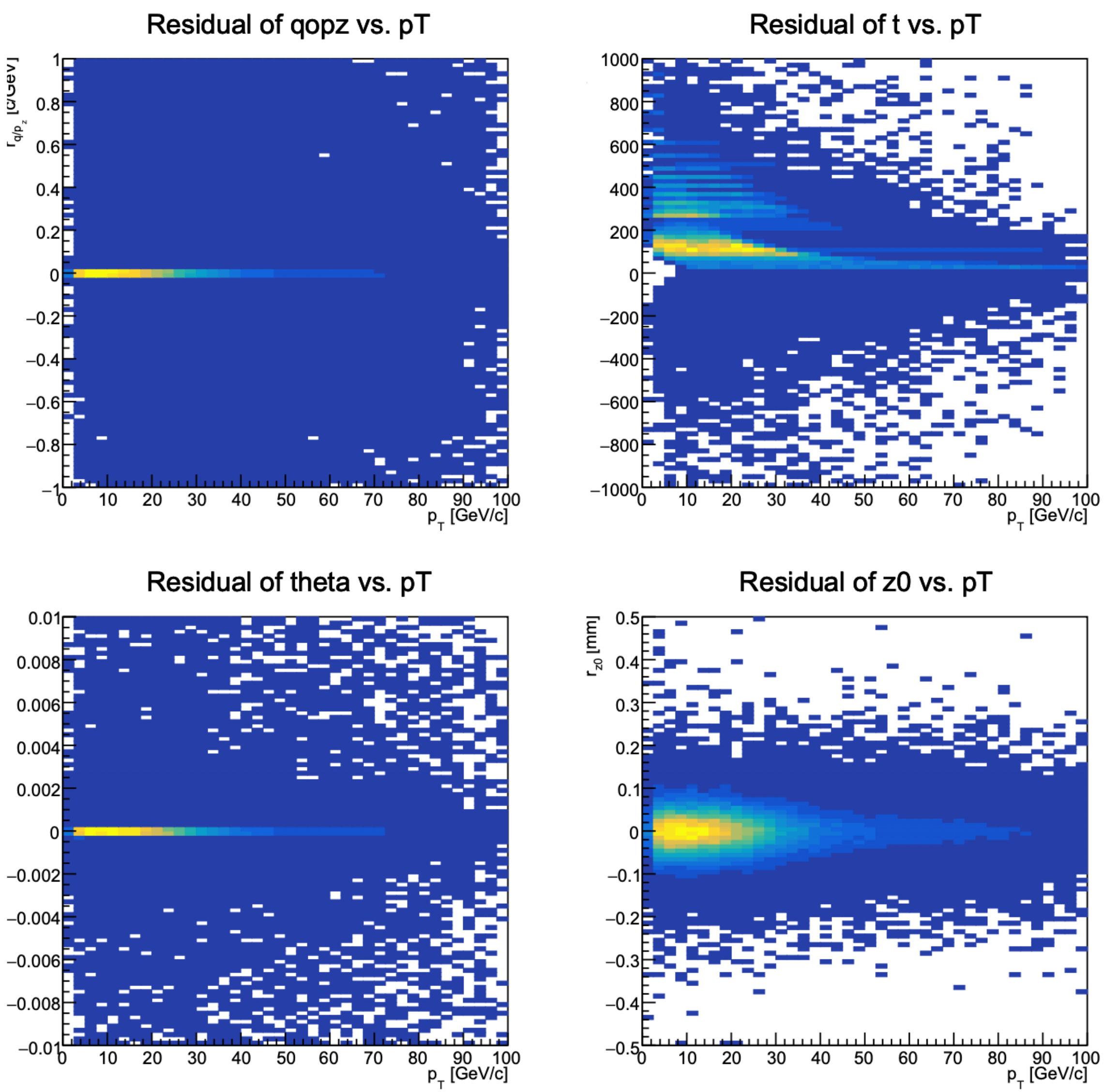
In KF, the deviation is about 1.6% on CUDA, about 0.7% on CPU.



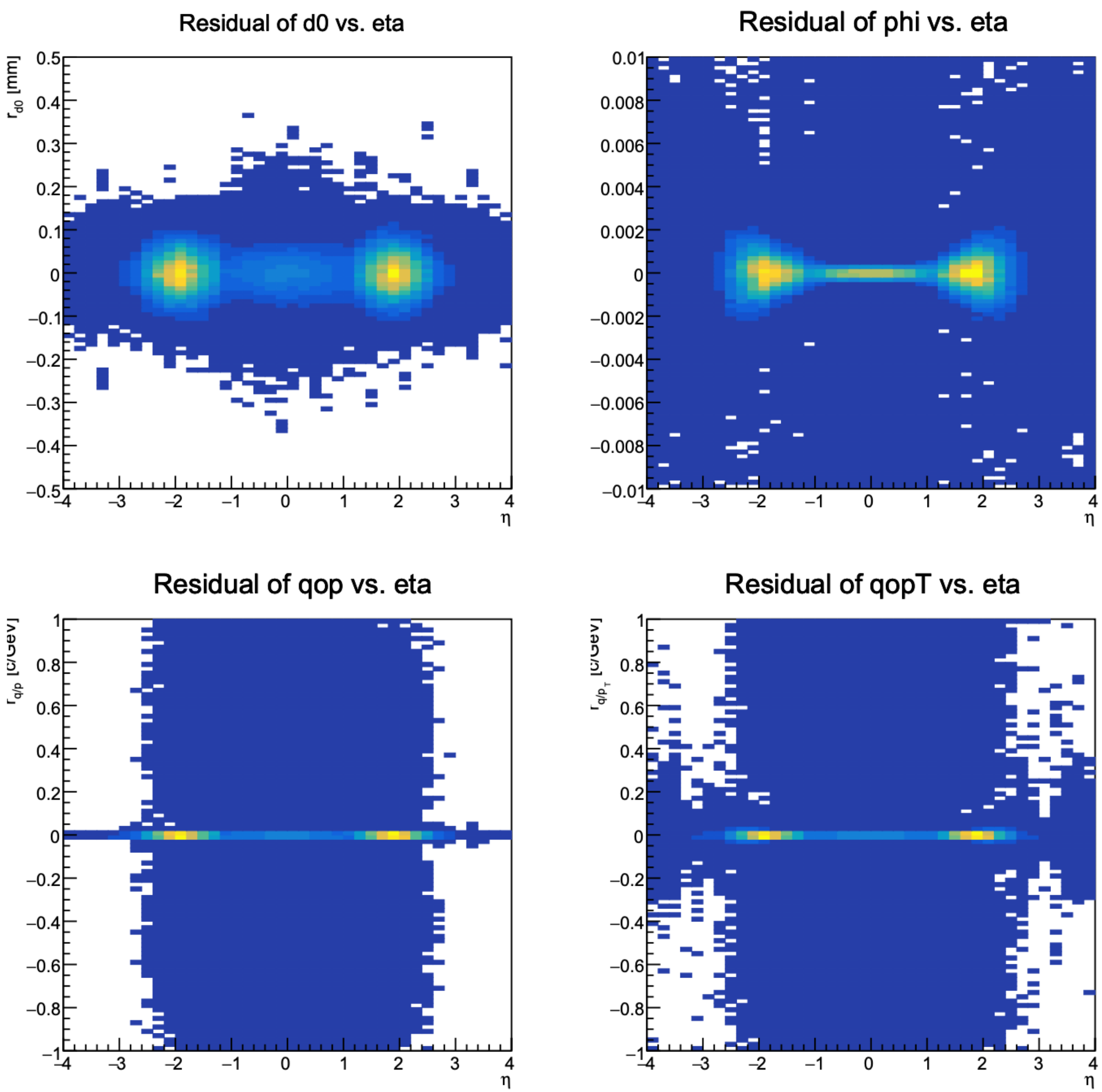
Residuals of each parameter 1/4



Residuals of each parameter 2/4



Residuals of each parameter 3/4



Residuals of each parameter 4/4