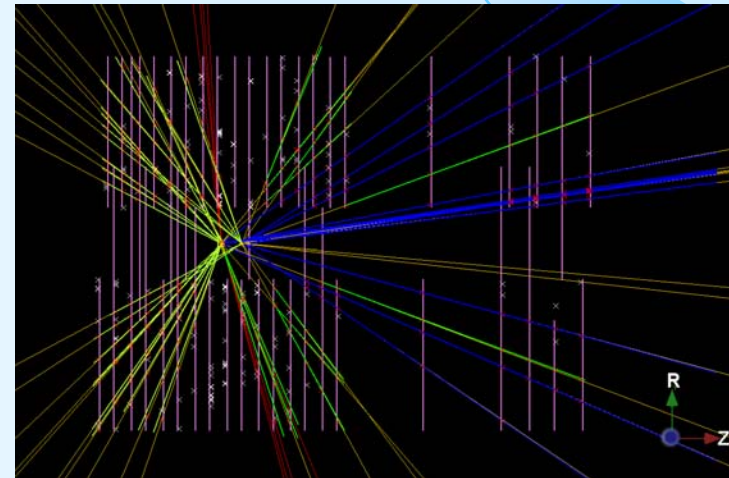


Refitting Tracks from DST

E. Rodrigues, NIKHEF

LHCb Tracking and Alignment Workshop, Lausanne, 8-9th November 2006

- ❖ *Motivations*
- ❖ *Step-by-step ...*
- ❖ *Current Performance*
 - *Tests with Brunel v30r12*
- ❖ *Outlook*



Reconstruction and Physics

- **Refitting tracks is a necessity in real world analyses**
 - e.g.: needed after improvement in knowledge of alignment

Event Model

- **One of the reasons for the track event model review in 2004-5**
 - *Nobody knew how to refit tracks from DST with the old event model*
- **Ability to refit was a request to us**

... WAS STILL AN UNTACKLED ISSUE ...

In Brunel ...

- **Whole of the tracking sequence:**
 - ❖ *Pattern recognition*
 - ❖ *Track fitting*
 - ❖ *Clone killing*
 - ❖ *tracks stripped of non-persistent information*
 - *left basically with LHCbIDs and flags*
- **job done with the “TrackToDST” algorithm**

... saved on the DST

- **Container of “BEST” tracks**

Two “kinds” of refitting

I. Refit a single track

- *probably mostly useful for tests, special studies, etc.*

- *Note: can also be done in Python ... can show interesting things ...*

II. Refit all tracks in a container

- *the common situation, I guess*

□ Solving situation II implies solving I

- *options to refit a track simply applied on the set of container tracks*

Refitting tracks from DST

- **Read the cocktail of tracks in the “best” container**
 - **Separate them back accordingly, in their original containers**
 - *Rec/Track/Forward, Rec/Track/Match, etc.*
 - **Refit per container with appropriate options**
 - *options depend on the track type!*
- **Need for a new algorithm ...**
- ✓ ***New TrackFromDST algorithm in Tr/TrackUtils***
 - classifies the tracks given as input according to their History / pattern recognition algorithms, i.e. remakes the PR algorithms produced in Brunel
 - BUT: all clones are gone! Not quite the same containers ...

Home-made DST

- ❖ **Minimalistic Brunel job – only tracking**
- still done in Brunel v30r10

Refit tracks from DST

- ❖ **Can do study in (at least) 2 ways:**
 - **in DaVinci**
 - **with a standalone / simple Gaudi job**
 - **opted for a standalone Python script**
- ❖ **New *DaVinciRefitting.opts*:**
options file with “refit” process phase (details given later)
- ❖ *own code to match “Brunel original’s” to refitted tracks*
- ❖ *and then compare the outcome ...*

Looking first at Long tracks from PatForward

- ❖ Same fitting options as in Brunel!
- ❖ Got many messages of the kind:

FitForward.Fitter.RefInfoTool WARNING LongTrackReferenceCreator:: No Velo State StatusCode=FAILURE

Tool to set reference info. on « Long » tracks

- ❖ *At the end of the job:*

```
FitForward      INFO ===== TrackEventFitter Summary =====
FitForward      INFO      Fitting performance      :      99.60 %
FitForward      INFO                               (  9604 /  9643 tracks )
FitForward      INFO =====
```

Conclusions:

- ❖ *All these tracks had been successfully fitted in Brunel*
- by construction, since they were taken from the “best” container
- ❖ *This first result is encouraging, but not good enough ...*
- ❖ *Let’s have a look at the reference information tools ...*

✓ **Tr/TrackTools**

➤ **LongTrackReferenceCreator:**

- *Adapted so that it can be used for tracks from DST, i.e. without “EndVelo” and AtT” states – these are only available in Brunel*
- *... now it takes closest states*
 - *improvements do not influence behaviour in Brunel*

Reference information tools for refitting:

- ❖ *In Brunel we have for now tools for only Long and seed (Tsa) tracks*
- ❖ *Tool for Long tracks successfully adapted/generalised*
 - *LongTrackReferenceCreator*
- ❖ *Tool for Tsa seed tracks cannot be used for now outside Brunel*
 - *Accesses pattern recognition intermediate information*
 - *We could/should make the tool flexible for refitting applications*
 - *Or make a new dedicated tool*
- ❖ *Tools for VeloTT and KShort fitting would be desirable ...*

The special case of Velo tracks:

- ❖ *In Brunel they are “prepared for fitting”*
 - *Get a P_T of 400 MeV*
 - *P set accordingly, depending also on slope*
- ❖ *Are prepared only those Velo tracks that are not ancestors of tracks in the “best” container*
- ❖ *For other purposes/tests it can be handy not to take into account the “best” container ...*
- ✓ **Tr/TrackUtils**
 - **TrackPrepareVelo:**
 - ❖ *modified so that Velo tracks ancestors of tracks in the “best” container can also be “prepared”*

- For now all in a file: *DaVinciRefitting.opts*
 - ❖ “Refit” process phase
 - ❖ *Loading of all necessary libraries*
 - *tracking-related libraries*
 - *libraries for general needed tools (e.g.: cluster position tools)*
 - ❖ *Remake all original containers from the “best” container*
 - ❖ *Decoding of all tracking detectors – needed for track fit*
 - ❖ *Refitting of all containers*
 - *with dedicated options according to track type*
 - *same options as in Brunel (except for ref. tools – see comments before)*

Some words on technical details

- **Reference information set only for Long tracks**
- *tracks from PatForward and TrackMatching*

```

FitForward      INFO ===== TrackEventFitter Summary =====
FitForward      INFO   Fitting performance   :   100.00 %
FitForward      INFO                        (   9643 /   9643 tracks )
FitForward      INFO =====

FitMatch        INFO ===== TrackEventFitter Summary =====
FitMatch        INFO   Fitting performance   :   100.00 %
FitMatch        INFO                        (   6461 /   6461 tracks )
FitMatch        INFO =====
  
```

Much better !

- **Testing on other track types as well ...**

```

FitVeloTT      INFO ===== TrackEventFitter Summary =====
FitVeloTT      INFO   Fitting performance   :   99.86 %
FitVeloTT      INFO                               ( 2138 / 2141 tracks )
FitVeloTT      INFO =====
FitTsaSeed     INFO
FitTsaSeed     INFO ===== TrackEventFitter Summary =====
FitTsaSeed     INFO   Fitting performance   :  100.00 %
FitTsaSeed     INFO                               ( 11000 / 11000 tracks )
FitTsaSeed     INFO =====
FitKShort      INFO
FitKShort      INFO ===== TrackEventFitter Summary =====
FitKShort      INFO   Fitting performance   :   99.91 %
FitKShort      INFO                               ( 6483 / 6489 tracks )
FitKShort      INFO =====
    
```

- ❑ **Also encouraging**
 - ❖ some fit failures left to be investigated later ...

Home-made DST

- ❖ **Minimalistic Brunel job – only tracking**
- ❖ **Now done in Brunel v30r12**
- ❖ **Extended DST:**
has linker tables to be able to re-associate tracks

Refit tracks from DST

- ❖ **Extended standalone Python script**
- ❖ **same *DaVinciRefitting.opts***
- ❖ ***own code to match “Brunel original’s” to refitted tracks***
- ❖ ***Tracks re-associated to MC truth***
- ❖ ***comparisons to same Brunel original track and MC truth***

Brunel

- ❖ Track's original value in Brunel

DV

- ❖ "DaVinci" value, i.e. value after track refitting

 $X (DV - Brunel)$

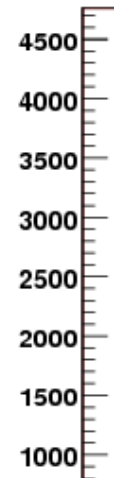
- ❖ Resolution in variable "X" – comparison refitting/fitting

Plots for x, y, t_x, t_y :

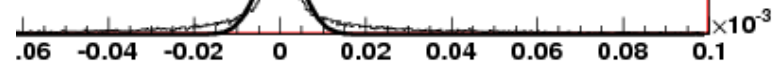
- ❖ Variable comparison looping over all states on the track

Very good agreement in slopes

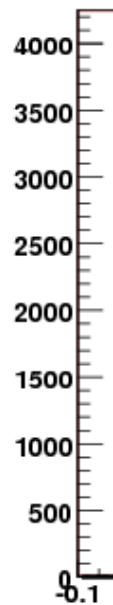
Forward Tx (DV-Brunel)



Forward: Tx at fitted states	
Entries	57474
Mean	2.293e-07
RMS	2.558e-05
Underflow	6550
Overflow	6599
χ^2 / ndf	2.258e+04 / 497
Prob	0
Constant	718.6 ± 18.8
Mean	4.056e-08 ± 3.278e-08
Sigma	4.829e-06 ± 1.225e-07



Forward Ty (DV-Brunel)



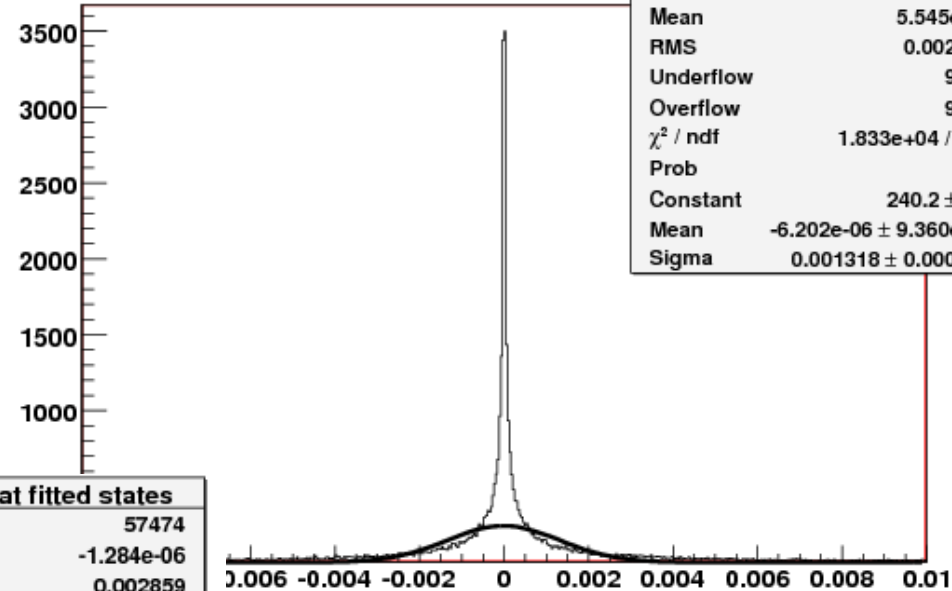
Forward: Ty at fitted states	
Entries	57474
Mean	-2.334e-08
RMS	2.671e-05
Underflow	7632
Overflow	7191
χ^2 / ndf	2.109e+04 / 497
Prob	0
Constant	334.4 ± 8.8
Mean	1.702e-07 ± 7.007e-08
Sigma	1.029e-05 ± 2.597e-07



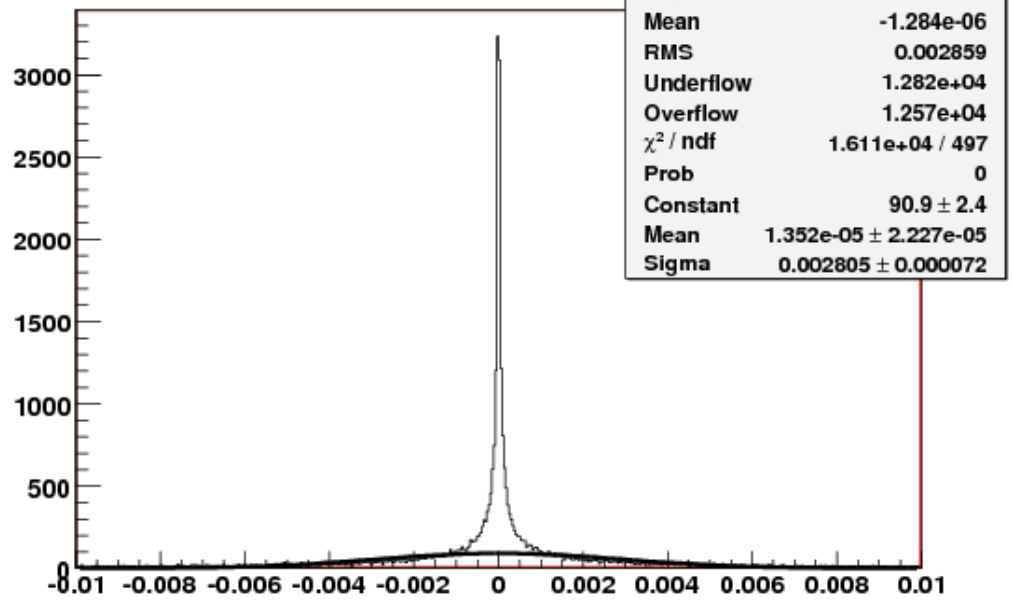
Tails to be understood

Very good Agreement in positions ~1-2 μm ... looping over Velo/TT/OT/... hits!

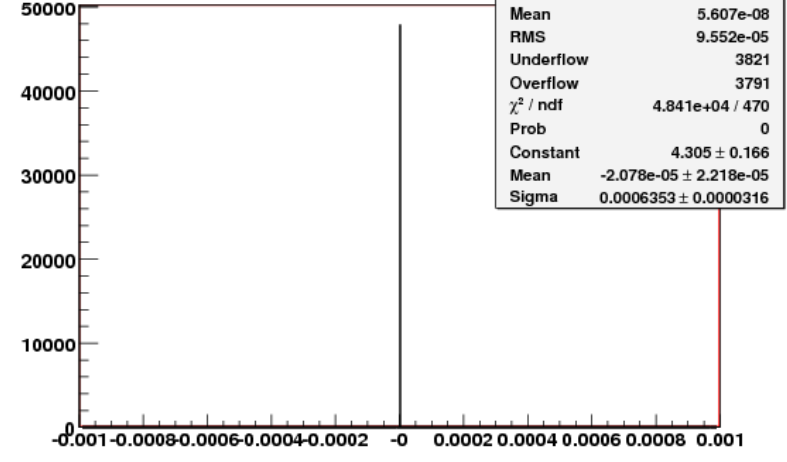
Forward X (DV-Brunel)



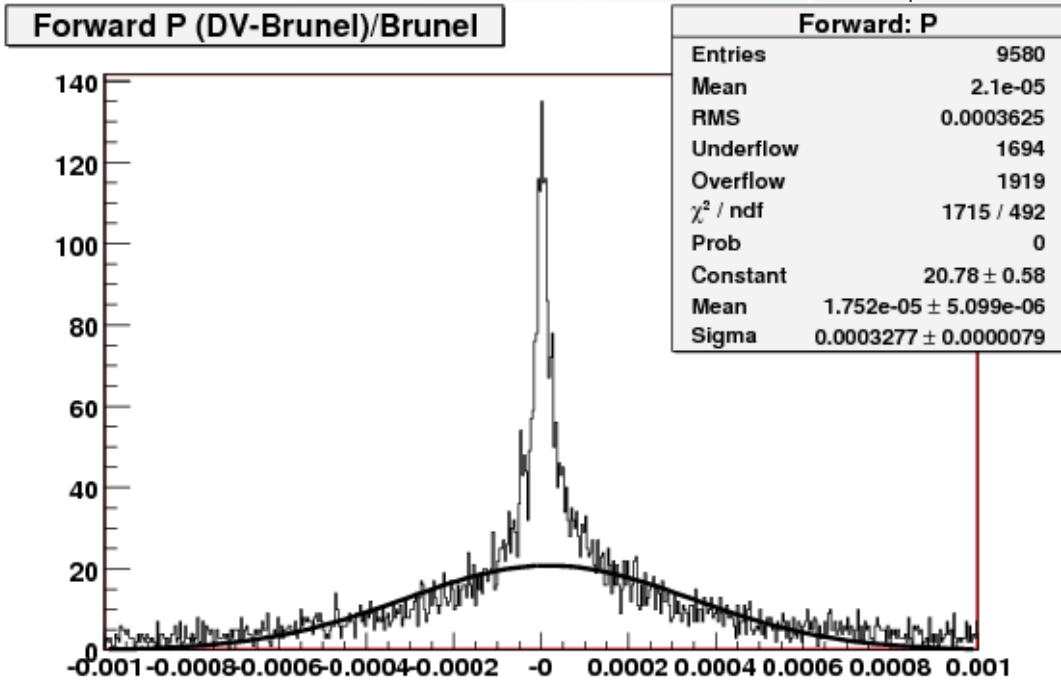
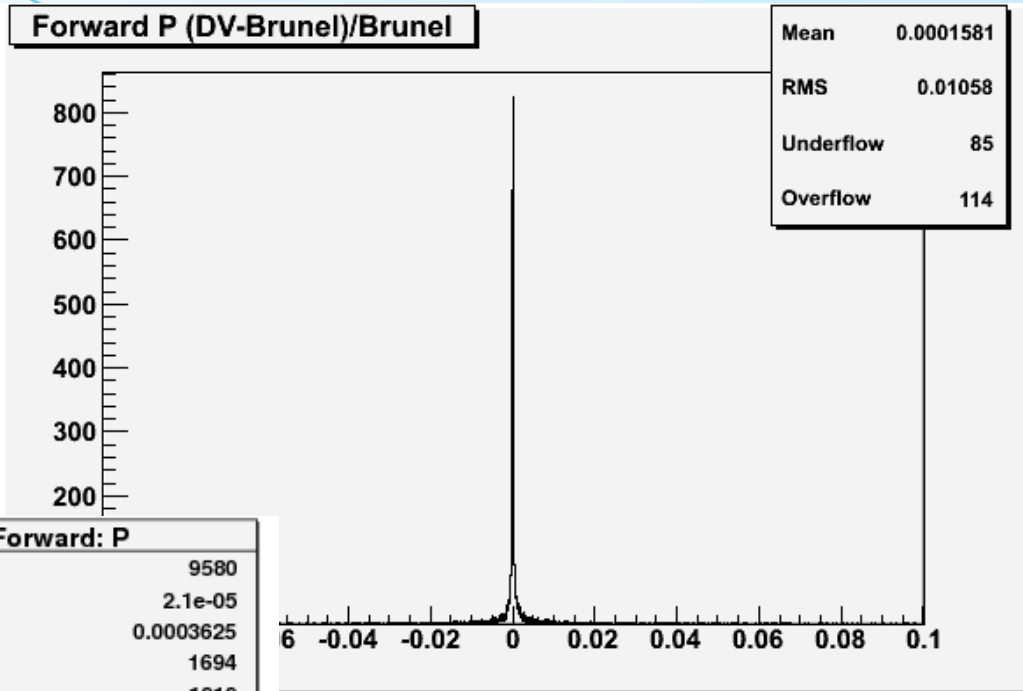
Forward Y (DV-Brunel)



Forward Z (DV-Brunel)

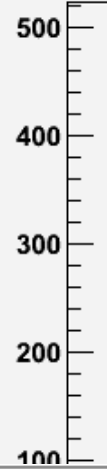


Very good agreement in momentum



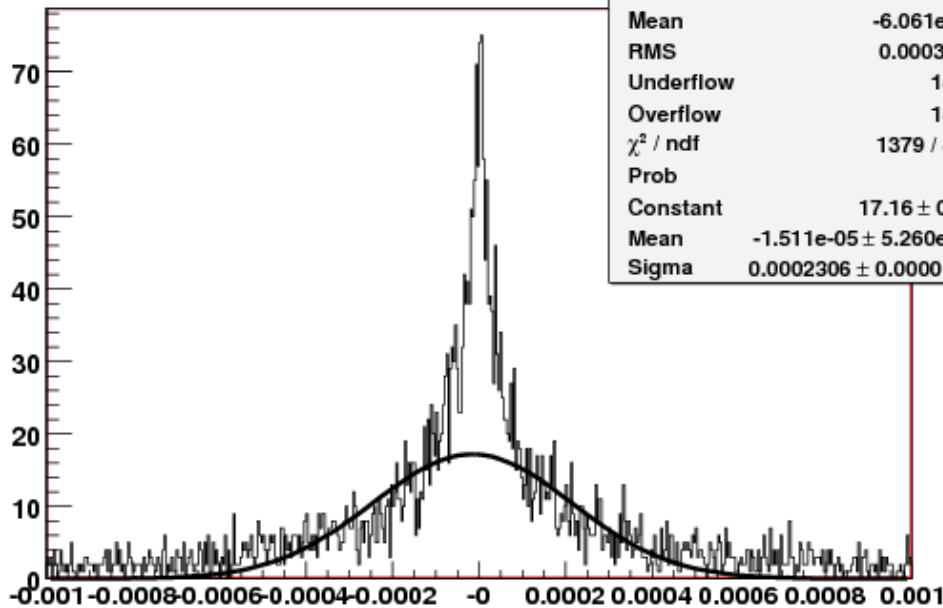
Very good agreement in momentum

Match P (DV-Brunel)/Brunel

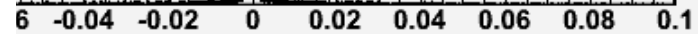


Mean	-0.0002516
RMS	0.01412
Underflow	95
Overflow	103

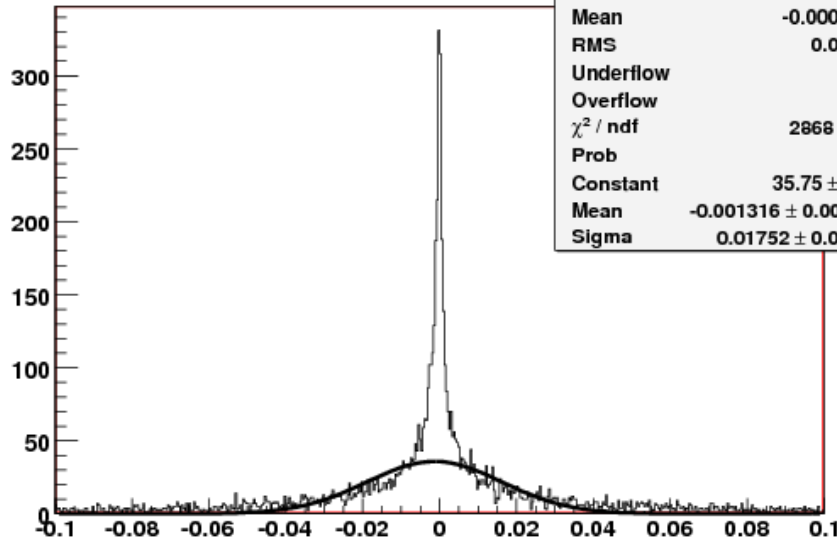
Match P (DV-Brunel)/Brunel



Match: P	
Entries	6691
Mean	-6.061e-06
RMS	0.0003451
Underflow	1445
Overflow	1392
χ^2 / ndf	1379 / 473
Prob	0
Constant	17.16 ± 0.98
Mean	-1.511e-05 ± 5.260e-06
Sigma	0.0002306 ± 0.0000125

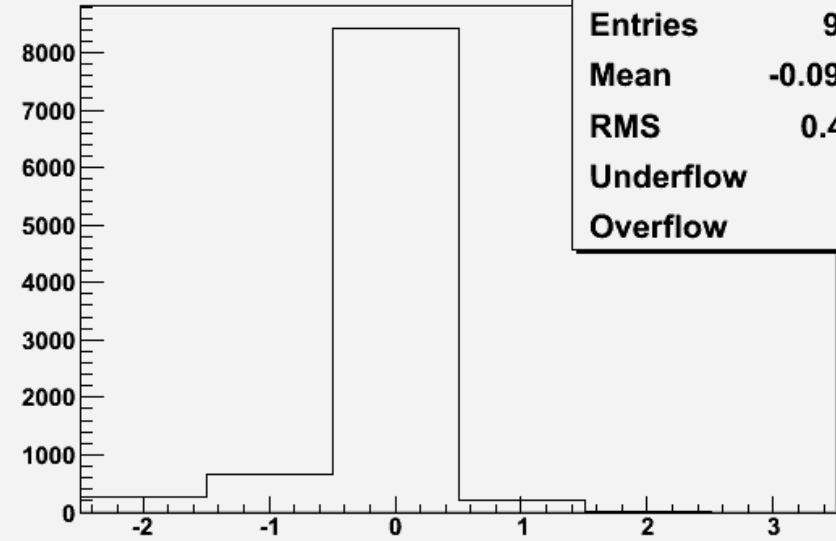


Forward Chi2/nDoF (DV-Brunel)/Brunel



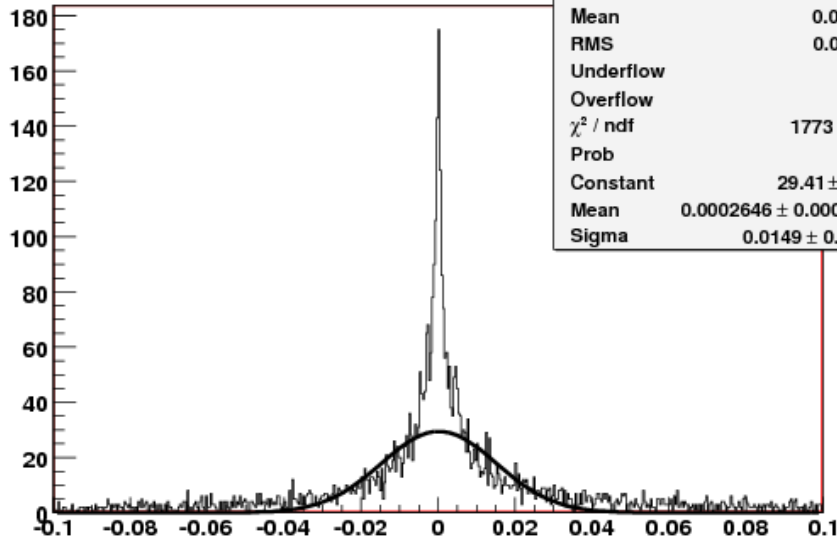
Forward: Chi2/nDoF	
Entries	9580
Mean	-0.0005577
RMS	0.03076
Underflow	1144
Overflow	1645
χ^2 / ndf	2868 / 487
Prob	0
Constant	35.75 ± 1.64
Mean	-0.001316 ± 0.000281
Sigma	0.01752 ± 0.00076

Forward nDoF (DV-Brunel)



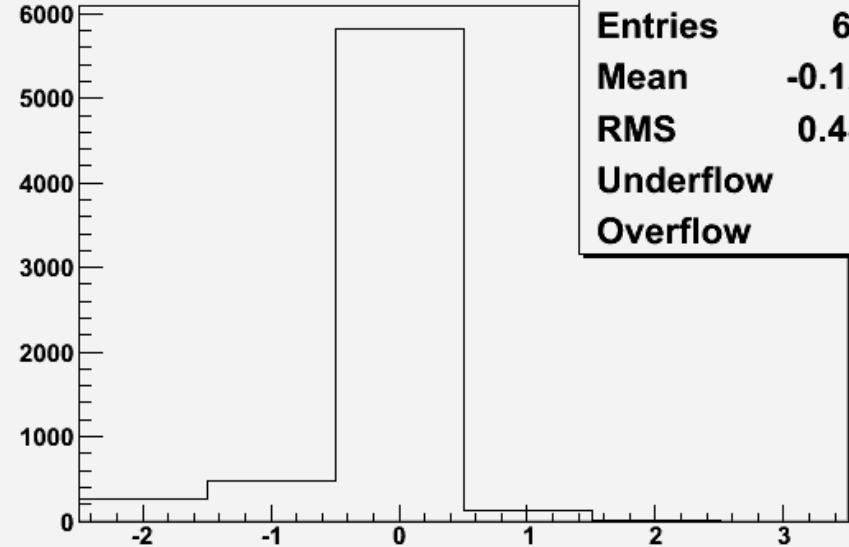
Forward: nDoF	
Entries	9580
Mean	-0.09656
RMS	0.4541
Underflow	0
Overflow	0

Match Chi2/nDoF (DV-Brunel)/Brunel



Match: Chi2/nDoF	
Entries	6691
Mean	0.00136
RMS	0.03102
Underflow	827
Overflow	1346
χ^2 / ndf	1773 / 465
Prob	0
Constant	29.41 ± 1.45
Mean	0.0002646 ± 0.0002871
Sigma	0.0149 ± 0.0007

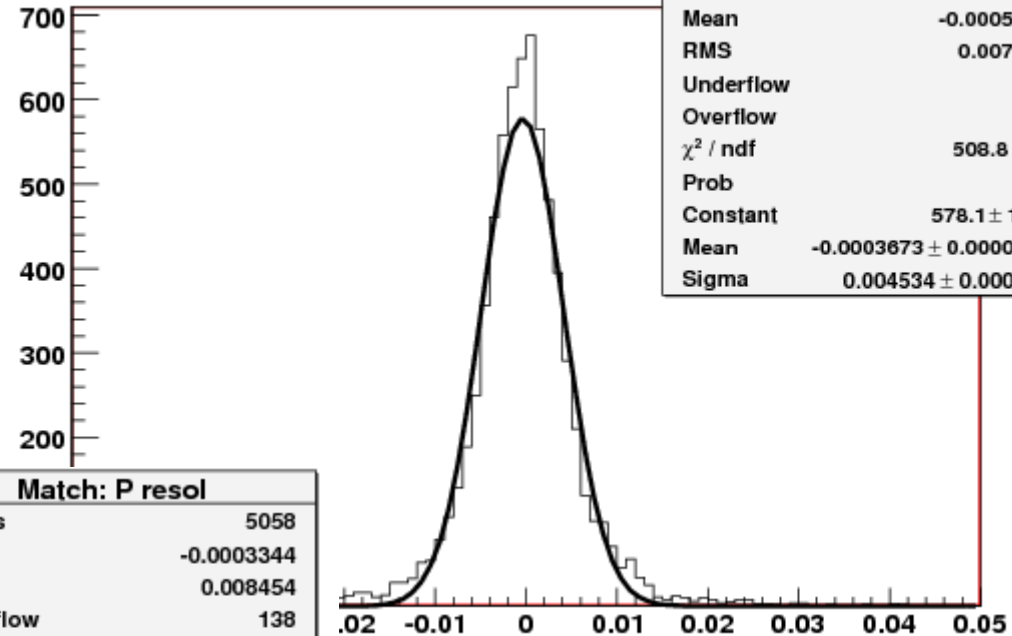
Match nDoF (DV-Brunel)



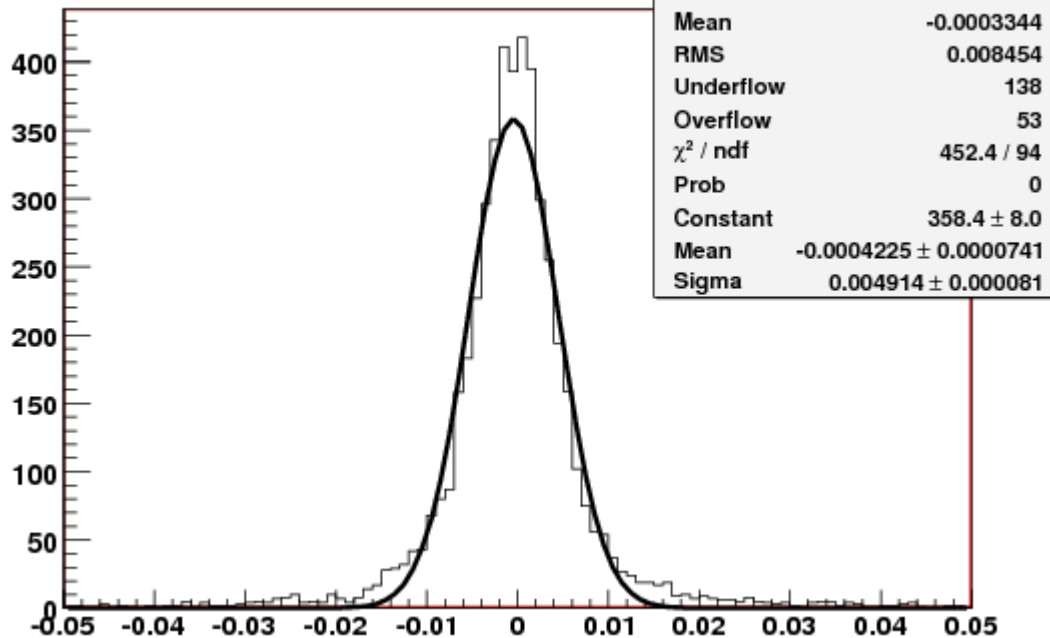
Match: nDoF	
Entries	6691
Mean	-0.1266
RMS	0.4896
Underflow	0
Overflow	0

Momentum resolutions as in Brunel!

Forward P resolution



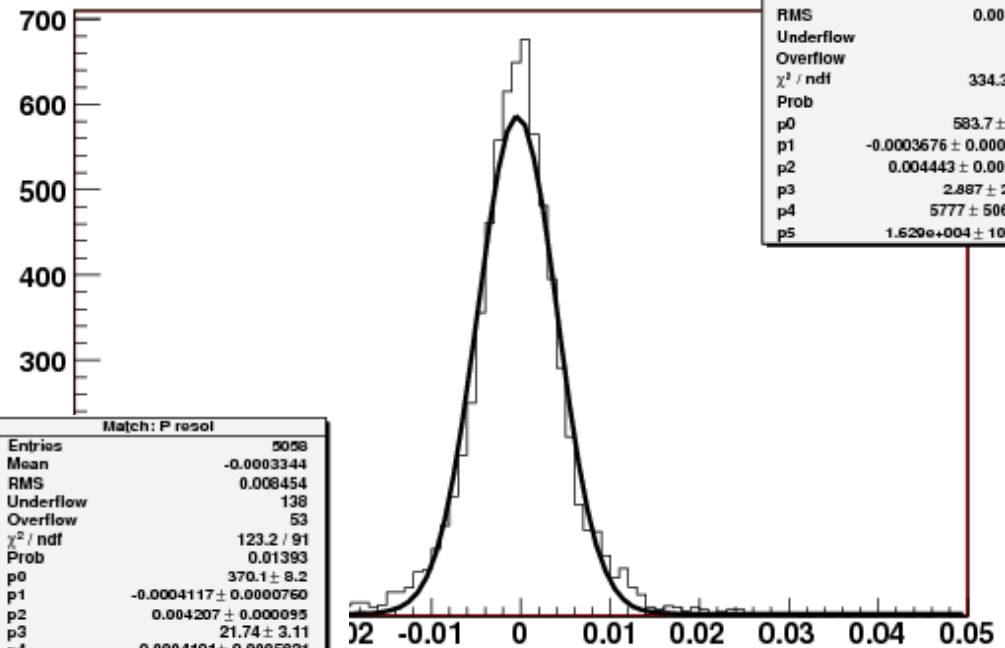
Match P resolution



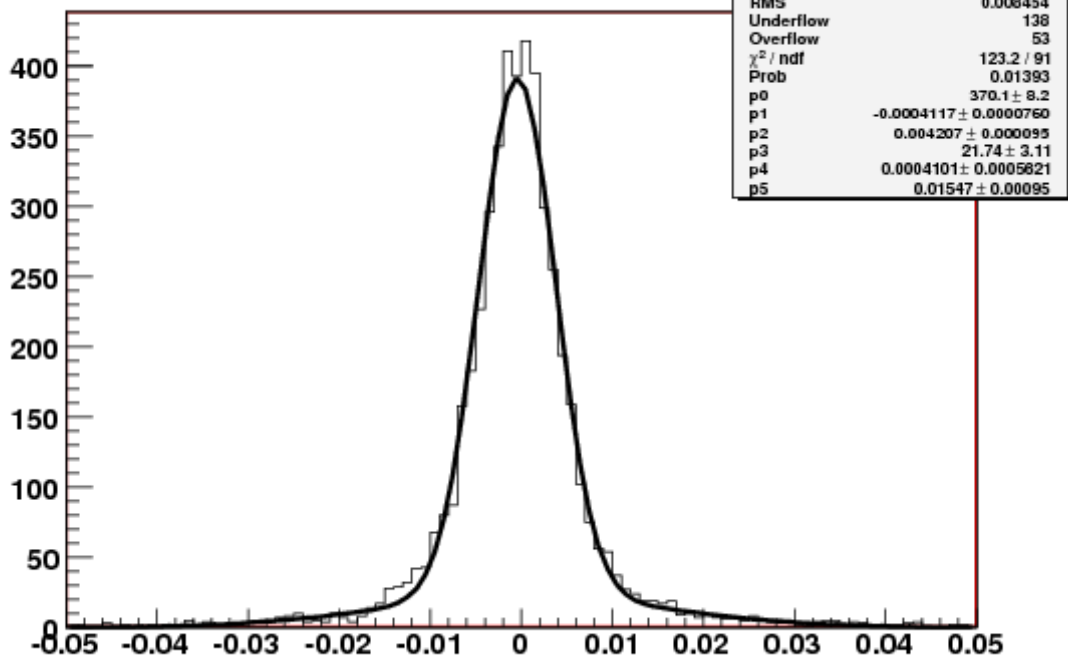
$\delta p / p \sim 4.5$ per mille

*Double Gaussian
Core resolutions
 $\delta p / p \sim 4.2$ per mille*

Forward P resolution



Match P resolution

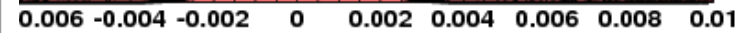


Very good agreement in positions

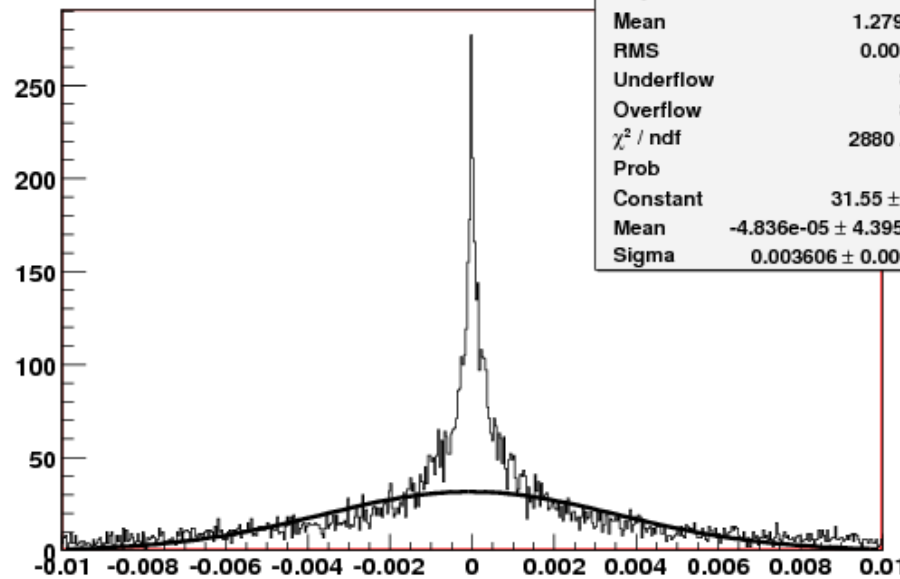
KsTrack X (DV-Brunel)



KsTrack: X at fitted states	
Entries	27084
Mean	1.695e-05
RMS	0.00287
Underflow	5583
Overflow	5490
χ^2 / ndf	7269 / 497
Prob	0
Constant	120.9 \pm 5.4
Mean	7.824e-06 \pm 1.235e-05
Sigma	0.001154 \pm 0.000050

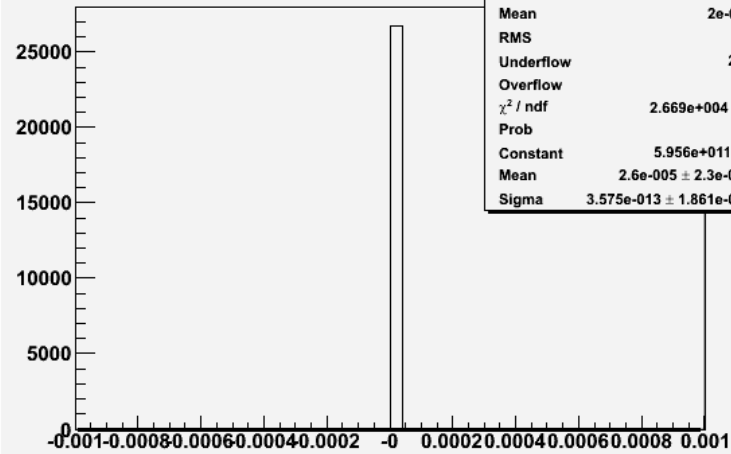


KsTrack Y (DV-Brunel)

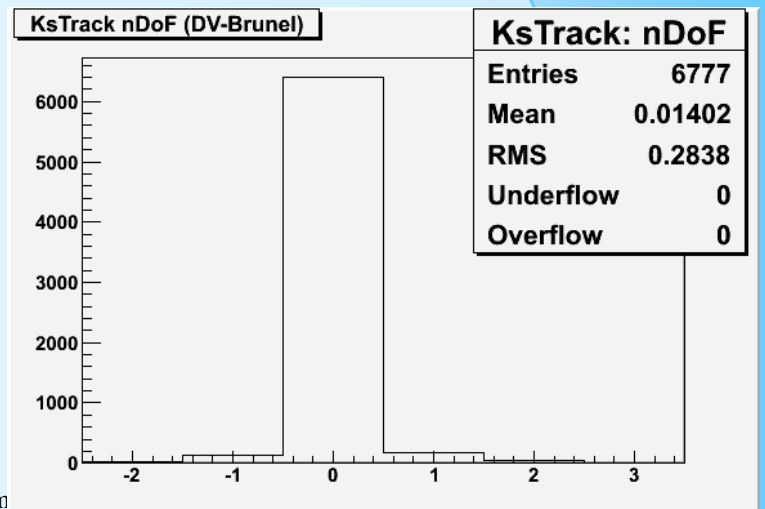
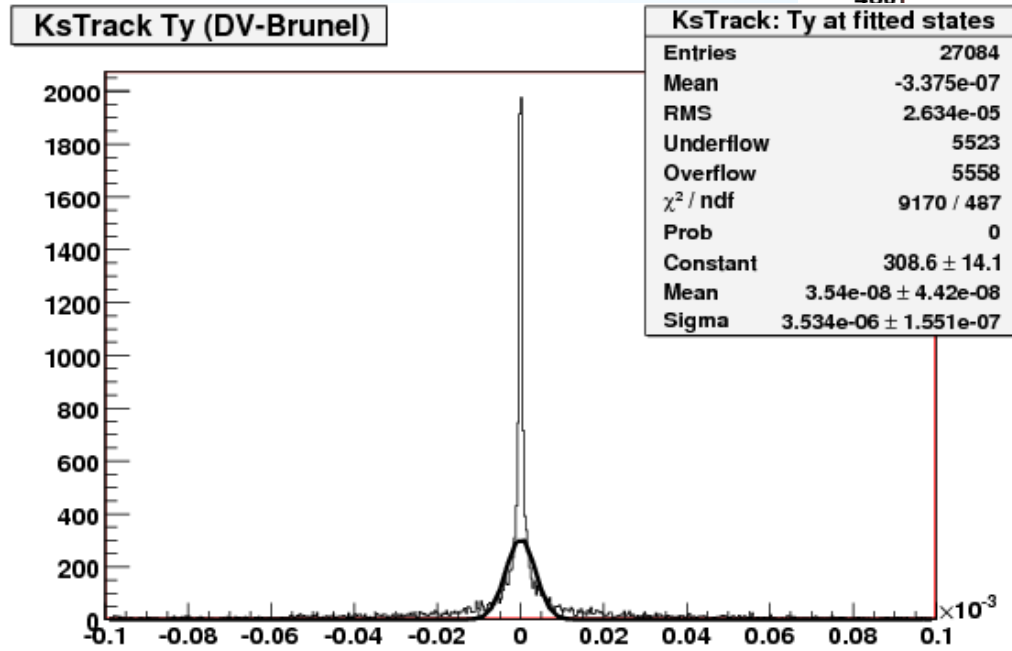
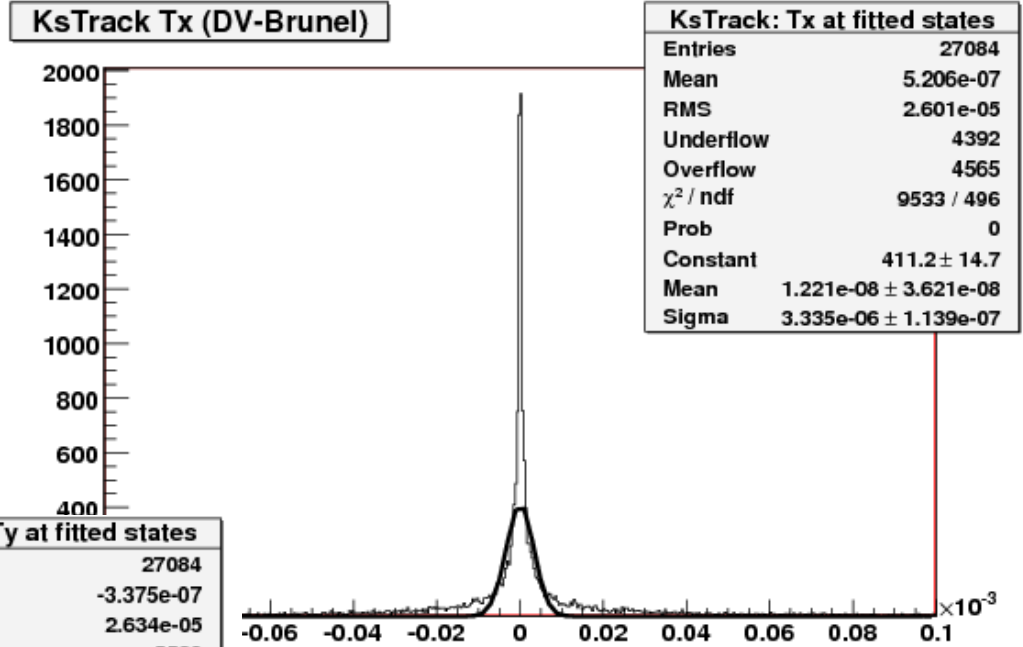
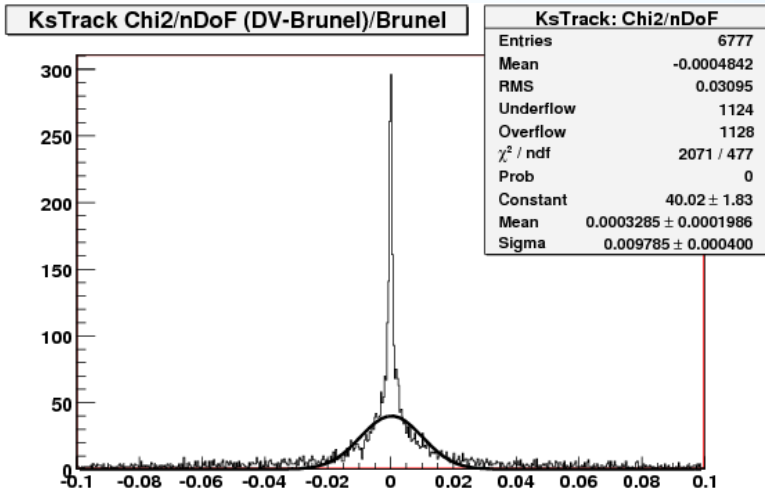


KsTrack: Y at fitted states	
Entries	27084
Mean	1.279e-05
RMS	0.003637
Underflow	8456
Overflow	8657
χ^2 / ndf	2880 / 497
Prob	0
Constant	31.55 \pm 0.67
Mean	-4.836e-05 \pm 4.395e-05
Sigma	0.003606 \pm 0.000067

KsTrack Z (DV-Brunel)

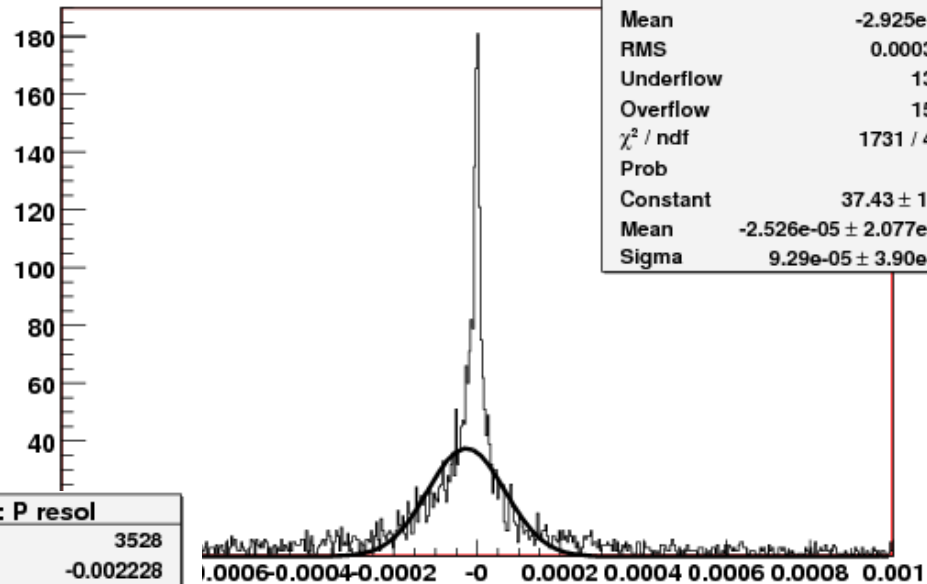


KsTrack: Z at fitted states	
Entries	27084
Mean	2e-005
RMS	0
Underflow	216
Overflow	181
χ^2 / ndf	2.669e+004 / -2
Prob	0
Constant	5.956e+011 \pm 1
Mean	2.6e-005 \pm 2.3e-015
Sigma	3.575e-013 \pm 1.861e-012

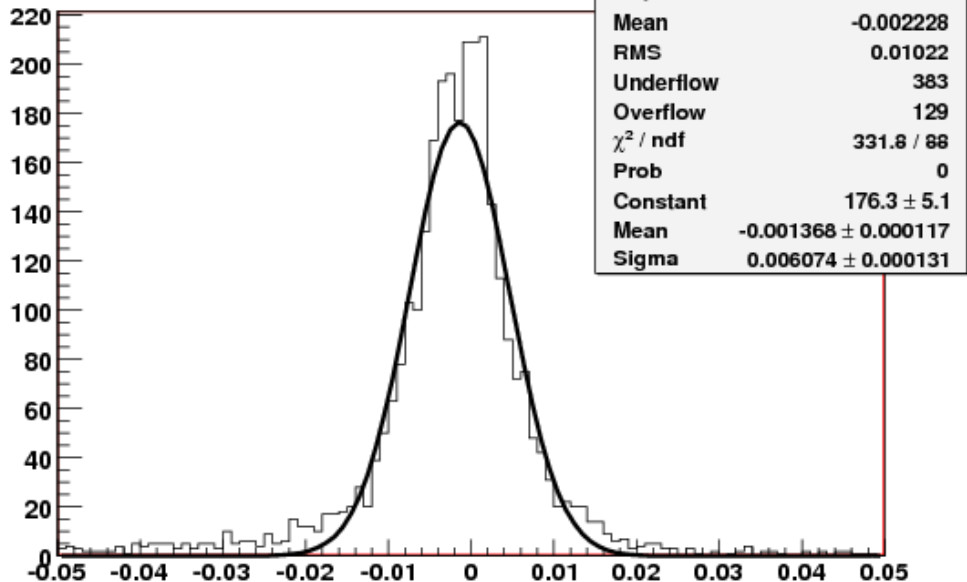


*Fair
agreement
in momentum*

KsTrack P (DV-Brunel)/Brunel



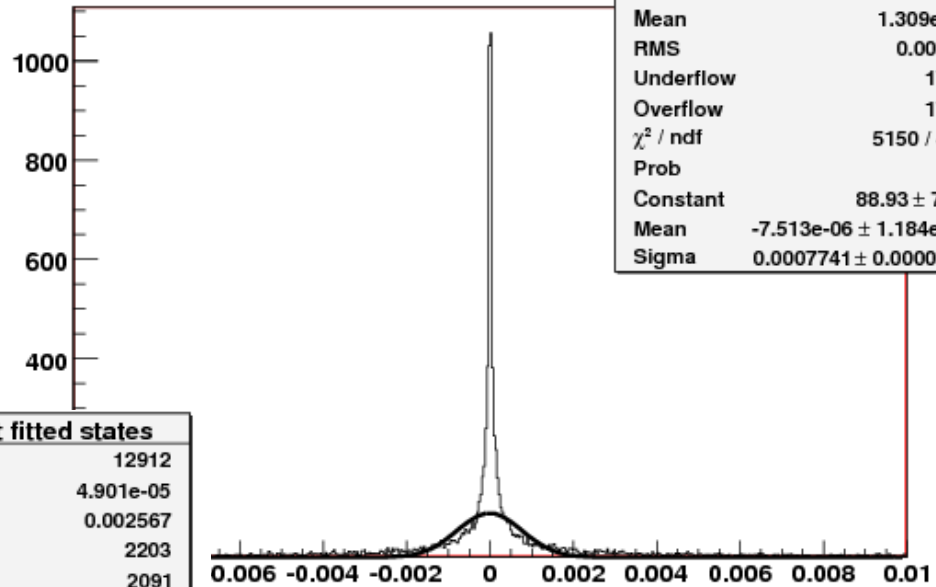
KsTrack P resolution



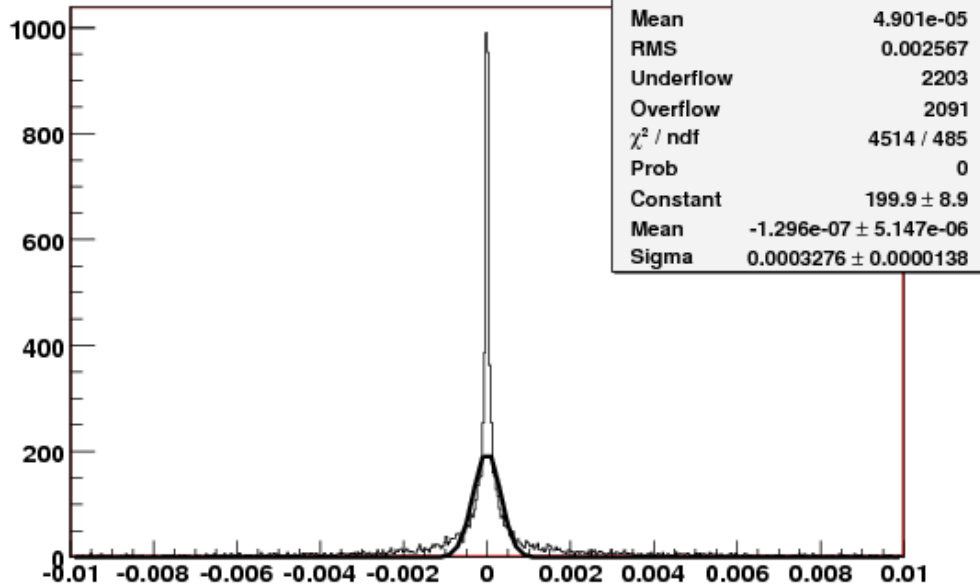
More fit iterations needed?

Too many outliers?

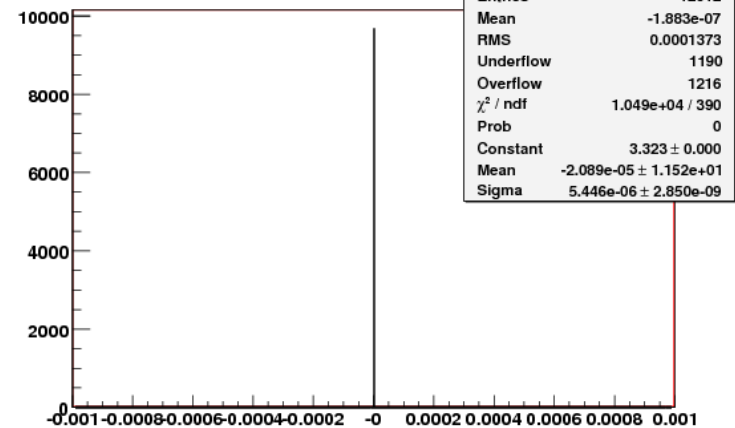
VeloTT X (DV-Brunel)



VeloTT Y (DV-Brunel)



VeloTT Z (DV-Brunel)



VeloTT Tx (DV-Brunel)

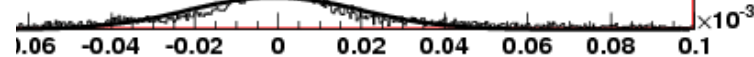


VeloTT: Tx at fitted states	
Entries	12912
Mean	-5.606e-07
RMS	2.95e-05
Underflow	1105
Overflow	1029
χ^2 / ndf	4652 / 496
Prob	0
Constant	48.15 \pm 2.36
Mean	-7.821e-07 \pm 2.656e-07
Sigma	2.03e-05 \pm 9.58e-07

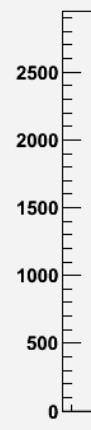
VeloTT Ty (DV-Brunel)



VeloTT: Ty at fitted states	
Entries	12912
Mean	-2.468e-07
RMS	2.91e-05
Underflow	1234
Overflow	1308
χ^2 / ndf	4612 / 492
Prob	0
Constant	58.77 \pm 2.07
Mean	-1.193e-09 \pm 2.063e-07
Sigma	1.563e-05 \pm 5.099e-07



VeloTT nDoF (DV-Brunel)



VeloTT: nDoF	
Entries	3257
Mean	0.07921
RMS	0.3606
Underflow	0
Overflow	0

- ✓ ***First proof that we can refit tracks from DST***
 - **Refitting is now a reality!**

- ✓ ***First tests are rather encouraging***
 - Differences between “Brunel-fitted” and “DST-refitted” tracks are in general well within errors
 - momentum resolutions of refitted tracks very much the same as in Brunel

- ***BUT one needs to understand:***
 - Tails in the various distributions
 - Fit failures for tracks that had been successfully fitted in Brunel
 - Refitting of all track types in detail