Offline Reconstruction Status Report

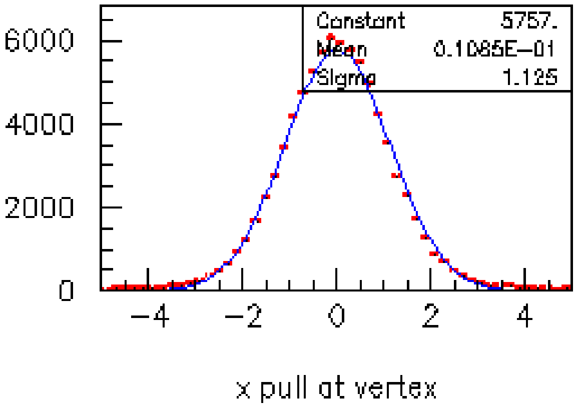
October 1, 2004
LHCb Week, Sardinia

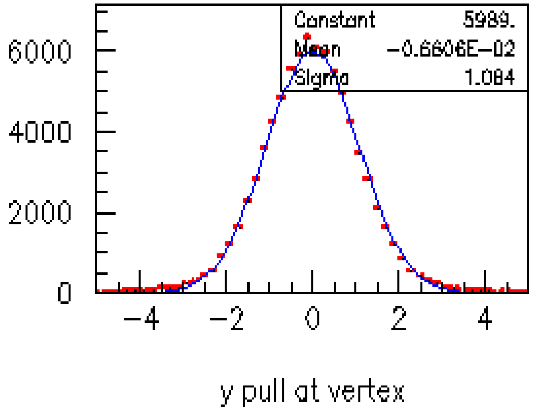
Status Report

- DC04 Reconstruction quality of tracks and vertices
  (plots and results provided by Yuehong Xie)

- New track reconstruction developments
  - New tracking event model - Jose Hernando and Eduardo Rodrigues
  - Track Seeding - Matthew Needham
  - A fast Kalman fit - Jeroen van Hunen

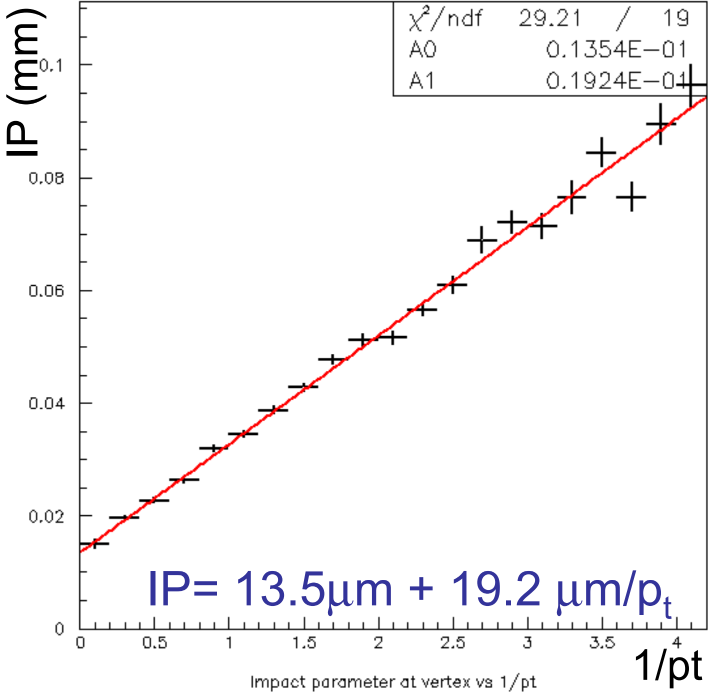
Long tracks: position reconstruction



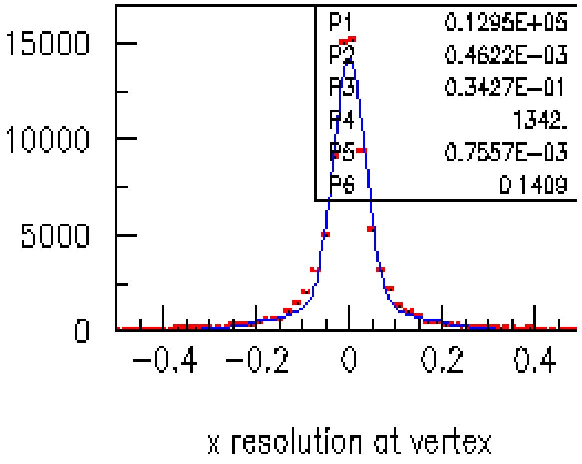


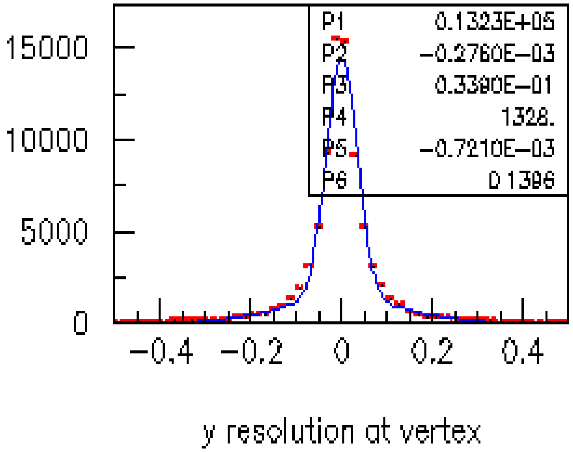
=> Pulls are OK for physics

Impact parameter resolution:



IP= 13.5µm + 19.2 µm/p_t

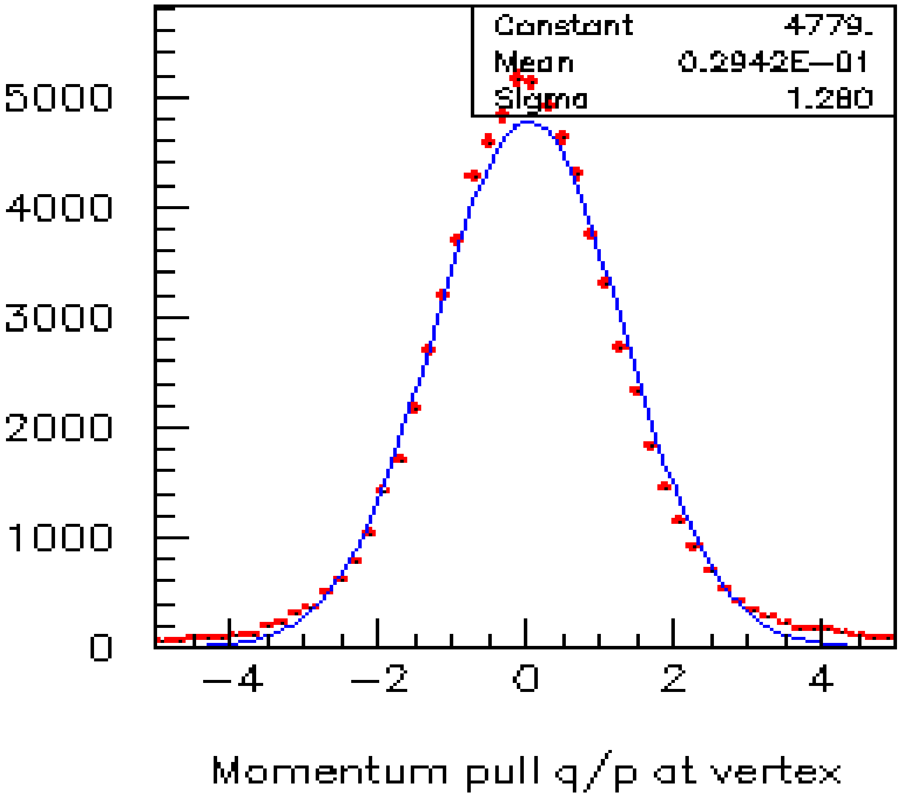


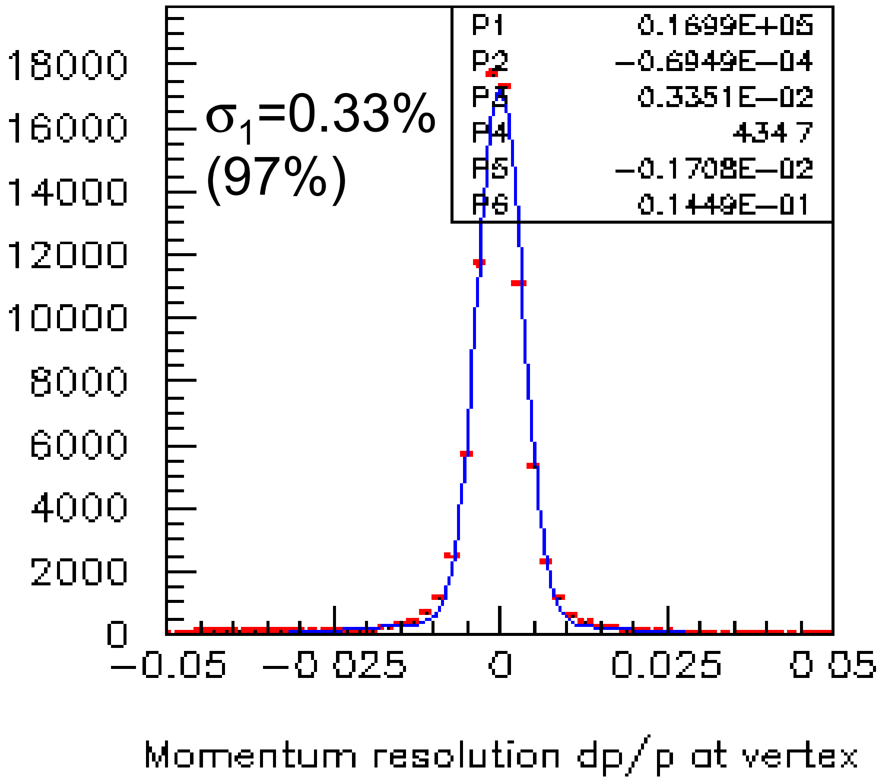


Resolution slightly better than TDR:

- better material description
- better velo cluster errors

Long tracks: momentum reconstruction



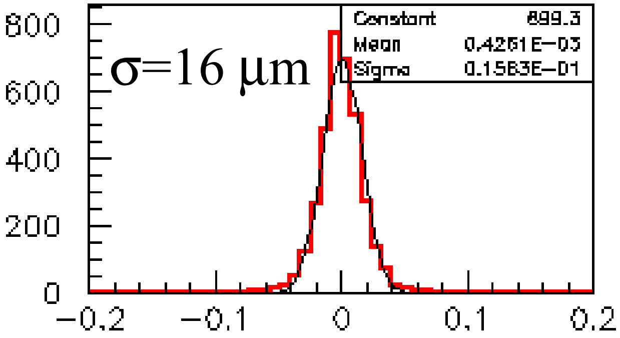


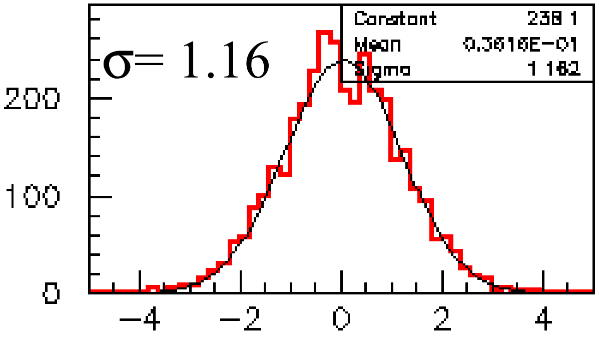
Momentum pull underestimated by 28% (same as in TDR):
- to be further investigated

Average momentum resolution slightly better than in Light TDR

2 prong B vertex: $B_d \rightarrow \pi\pi$

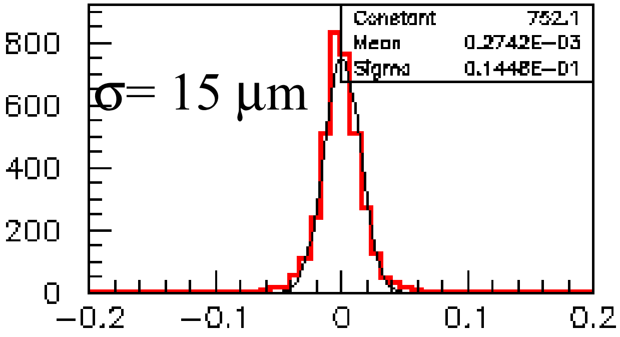
X res

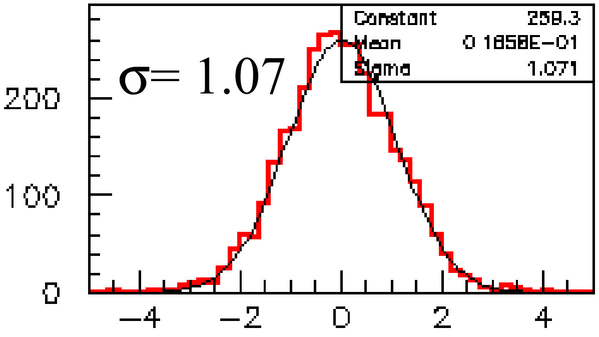




X pull

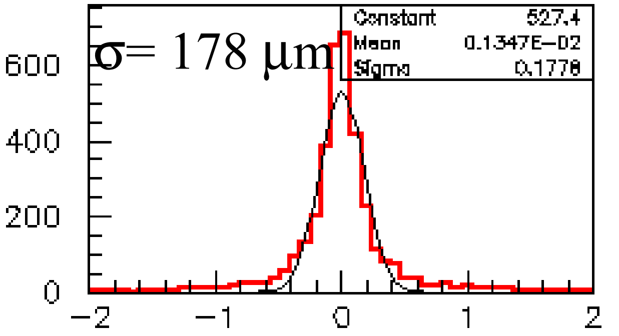
Y res

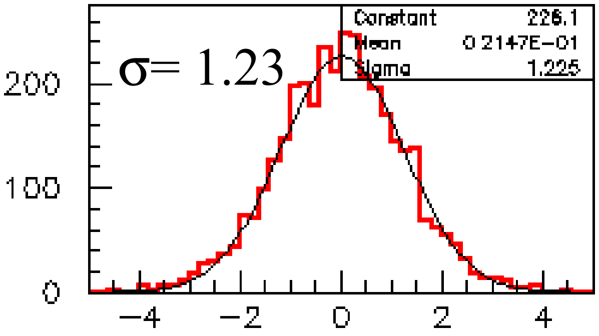




Y pull

Z res



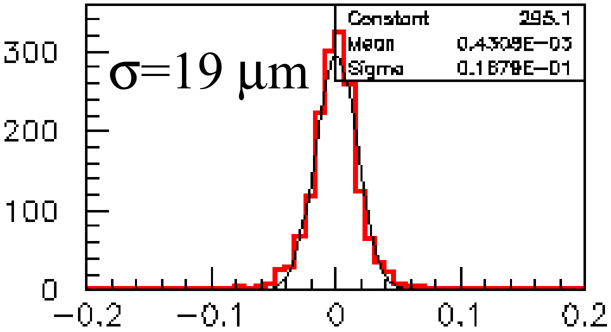


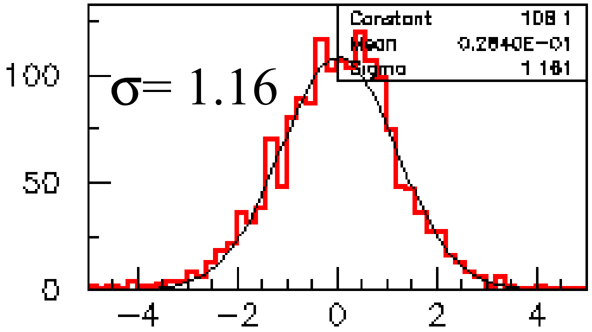
Z pull

OK

4 prong B vertex: $B_s \to D_s K$

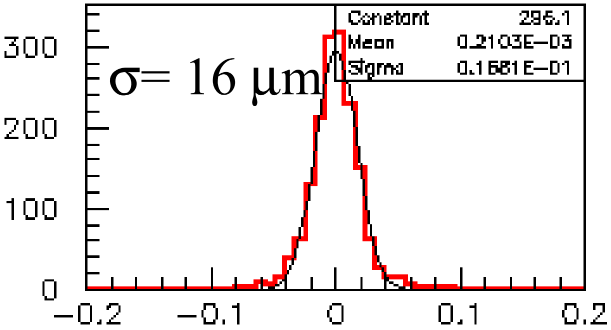
X res

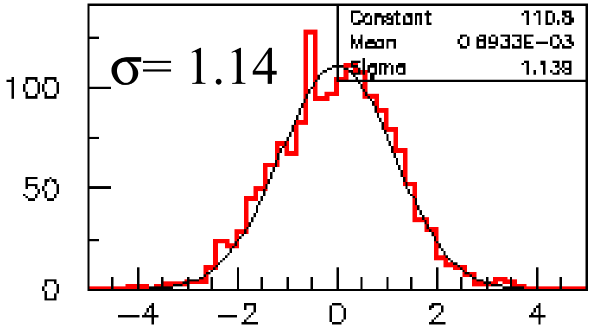




X pull

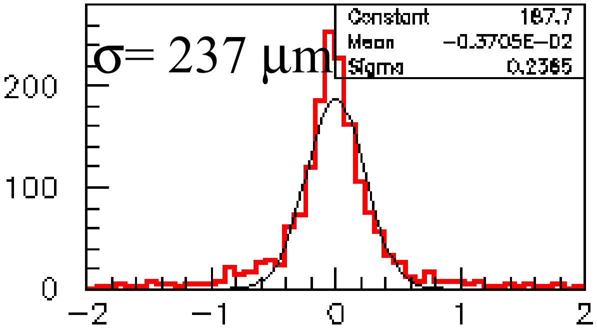
Y res

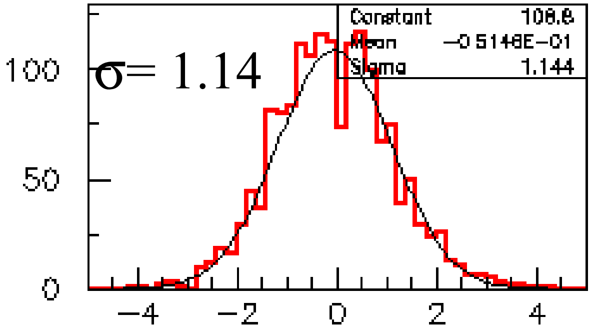




Y pull

Z res





Z pull

OK

All track types:σ of track parameter pulls

	x	y	tx	ty	dp/p
long	1.13	1.08	1.08	1.07	1.28
downstream	1.47	1.55	1.33	1.38	1.62
upstream	1.66	1.67	1.45	1.46	1.44
seed	2.18	1.55	2.05	1.51	1.99
velo	1.68	1.55	1.67	1.65	NA
veloBack	1.18	1.19	1.34	1.32	NA

Green: ok ; Orange: some systematics to be checked

Red : to be improved ; Purple: long extrapolation to track vertex

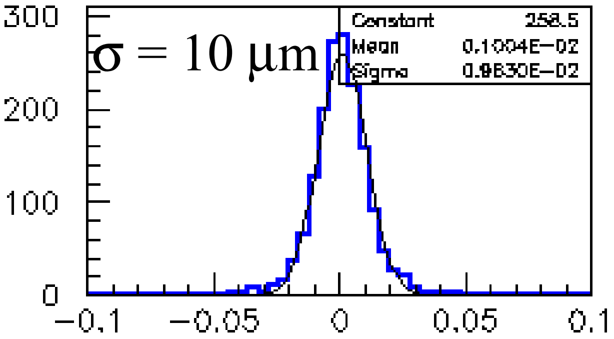
All track types: Core resolutions

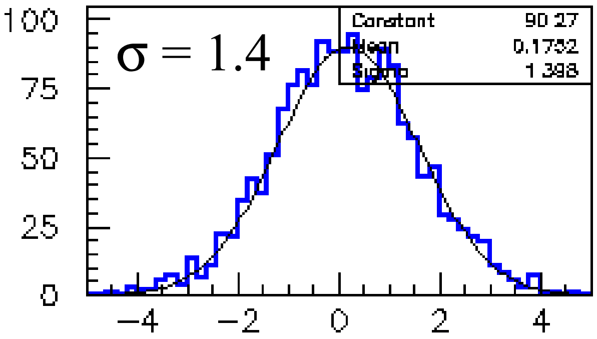
(for reference only)

	x	y	tx	ty	dp/p
long	.034	.034	.30e-3	.30e-3	.34e-2
downstream	/	/	.74e-3	.74e-3	.37e-2
upstream	.044	.044	.54e-3	.53e-3	.15
seed	/	/	.40e-3	.45e-3	.92e-2
velo	.047	.047	.65e-3	.50e-3	NA
veloBack	.041	.041	.58e-3	.56e-3	NA

Primary vertex

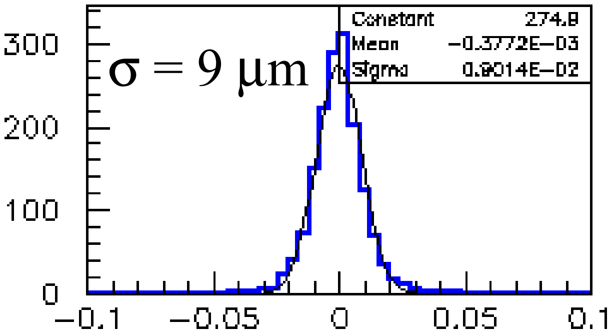
X res

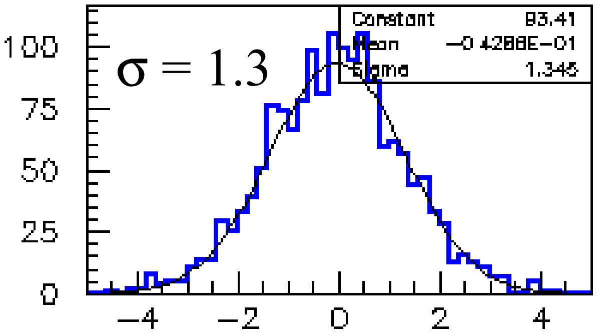




X pull

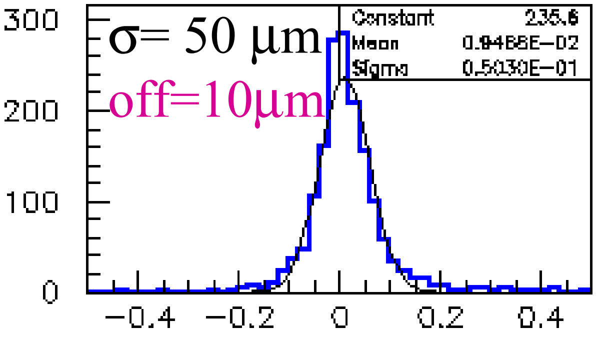
Y res

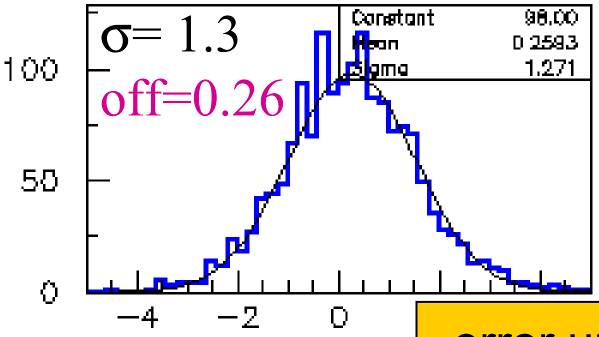




Y pull

Z res

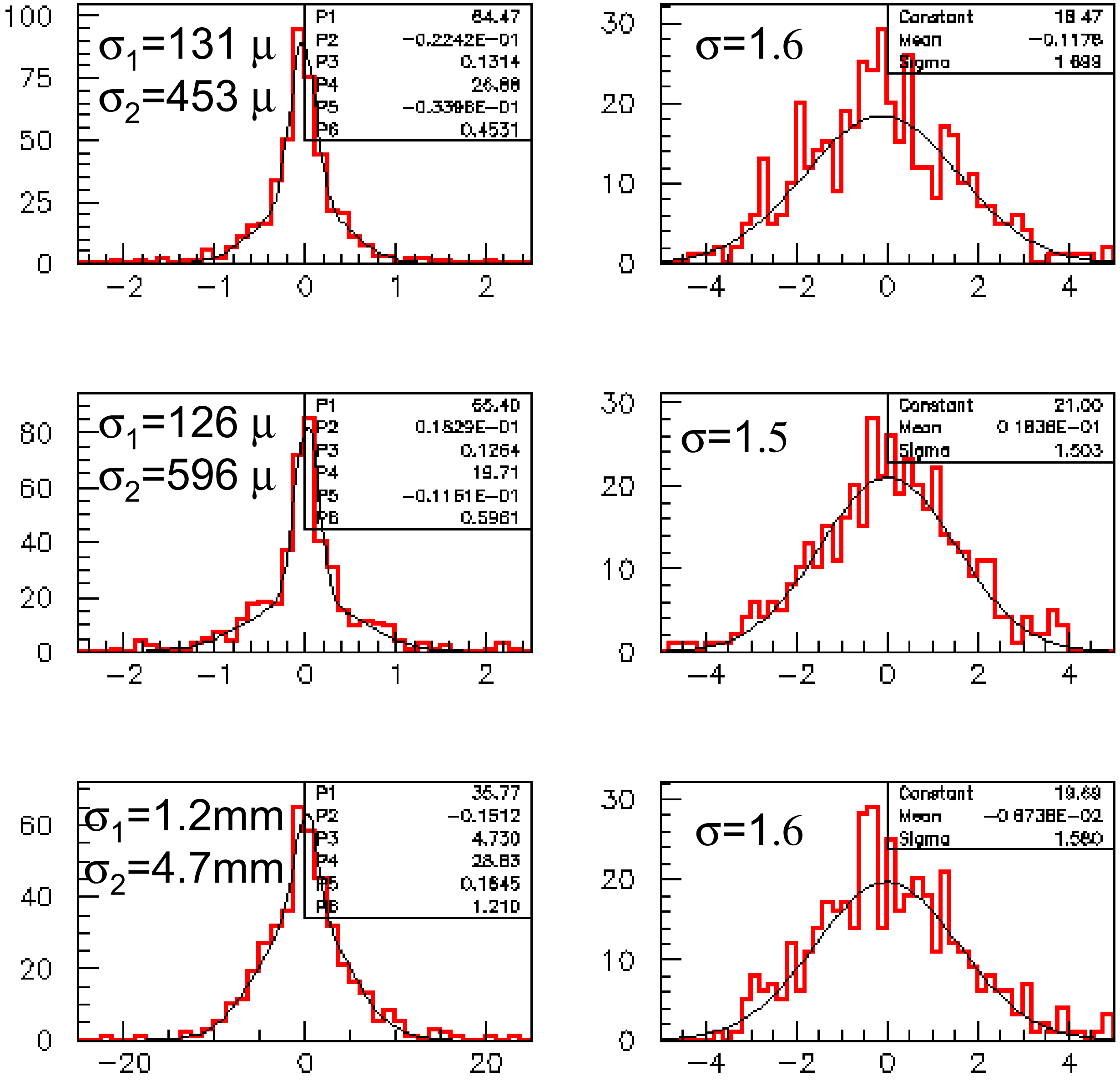




Z pull

-error underestimated
-small forward bias

B vertexing with Ks: Bs→KsKs



X res

X pull

Y res

Y pull

Z res

Z pull

Errors underestimated

Reconstruction quality summary

- Long tracks
  - IP and momentum resolutions slightly better than TDR
  - IP pulls OK, momentum pull to be perfected.
- Upstream and downstream tracks
  - Pulls have Gaussian shape, but errors are underestimated by 30% - 50%
- T tracks and Velo tracks
  - Errors at track vertex are not correctly modeled (factor ~2). To be improved.
- Effect on physics
  - Primary vertices are slightly forward biased in z (~10 μm)
  - B vertices with long tracks are OK
  - B vertices with Ks only have underestimated errors (50%-60%)

Status of new tracking data model

Jose Hernando (CERN) , Eduardo Rodrigues (NIKHEF)

History

- need for new model acknowledged ~ 1 year ago
- many discussions/mails/comments hold/exchanged/given over last months
  - ➥ O. Callot, M. Merk, M. Needham, T. Ruf, J. van Tilburg
- collection of "all" requirements
- set-up/implementation of new model under way
  - ➥ J. Hernando, E. Rodrigues
- complete proposal + implementation for end of this year

Main lines of thought

- common tracking base classes for trigger/offline
- common/generic/abstract set of tools
  - ➥ facilitates development of new algorithms
  - ➥ can be used by both trigger/offline reconstruction
- define input/output of track reconstruction
  - ➥ these should use the base classes
- standardize data that sub-detector algorithms can access from tracking
- standardize geometry access

TrTracks

What is it?

- a collection of states (likely that only first state made persistent)
- a collection of nodes (not to be made persistent)
- the quality of the agreement between track model and measurement
  - ➥ χ^2, # degrees of freedom

Is a track clever? NO!

- is does not know about detector geometry and alignment

What tracking code/experts could ask to a track?

- questions about all its attributes

What the end-user wants to know about a track?

- quality of track
- position/momentum/covariance at a certain point/plane/…
  - ➥ a “clever” and/or fast generic tool should provide this
  - ➥ user has choice on how fast/precise it wants the tool to be

(see tools later …)

What the end-user should not have to care about?

- how the job is done internally
- the particular position/details about the track states, …

TrStates

(only first – maybe second(?) – state is persistent)

What is it?

- vector of parameters defining a track trajectory at given points
- type
- an error covariance matrix

Is a state clever? NO!

- is does not know about detector geometry
- it does not know about alignment

What tracking code/experts could ask to a state?

- questions about all its attributes

What the end-user wants to know about states?

- the end-user should avoid using the states directly, if possible
  - ➥ ask questions to the track instead

What the end-user should not have to care about?

- the internal representation of the state

TrMeasurement

What is it?

- a measurement of a sub-detector associated to a track
- contains measurement + error
- contains type, flags(?), LHCbID (?)

Is a measurement clever? **NO!**

- is does not know about geometry
- it does not know about alignment

What tracking code/experts could ask to a measurement?

- questions about all its attributes

What the end-user wants to know about measurements?

- measurements are not relevant for the end-user

What the end-user should not have to care about?

- the end-user should only care about the final results of the fit

TrNode

(nodes are not persistent)

What is it?

- the link between the state and a measurement
- contains residual + error , pointer to measurement

Is a node clever? **Yes … could be …**

- the place to have access to geometry information
- could sort of hide the alignment since
  - ➥ a state should be in the general frame
  - ➥ a measurement should be in the local (i. e. sub-detector) frame

What tracking code/experts could ask to a node?

- questions about all its attributes

What the end-user wants to know about nodes?

- nodes are not relevant for the end-user

What the end-user should not have to care about?

- the end-user should only care about the final results of the fit

LHCbID

What is it? Could be …

- LHCbID = ID for each smallest piece of an LHCb sub-detector able to provide a measurement
- LHCbID = detector channel ID + bits to identify the sub-detector

Requirements

- can link to the Digits <-> also to RawBuffer
- can link to a list of MCParticles
- ability to access geometry
- has to be provided by reconstruction objects

➔ place where ideas/feedback/comments are (even more) welcome …

Tools - extrapolators

- at present these are more or less sophisticated tools deriving from ITrExtrapolator
- propose to expand all extrapolator tools to also provide position/momentum/covariance at a certain point and plane

e.g.:

```
/// Propagate a TrState to a given z-position
 virtual StatusCode propagate( TrState* state, double z = 0, ParticleID partId = ParticleID(211));
 /// Propagate a TrState to the intersection point with a given plane
 virtual StatusCode propagate( TrState* state, HepPlane plane, ParticleID partId = ParticleID(211));
 /// Retrieve the position and momentum vectors and the corresponding
 /// 6D covariance matrix (pos:1->3,mom:4-6) for a state at a given z-position
 virtual StatusCode positionAndMomentum( TrState* state, double z = 0,  ParticleID partId = ParticleID(211),
                                         HepPoint3D pos, HepVector3D mom, HepSymMatrix cov6D );
/// Retrieve the position and momentum vectors and the corresponding
 /// 6D covariance matrix (pos:1->3,mom:4-6) at the intersection of a state with a given plane
 virtual StatusCode positionAndMomentum( TrState* state, HepPlane plane, ParticleID partId = ParticleID(211),
                                         HepPoint3D pos, HepVector3D mom,  HepSymMatrix cov6D );
/// Retrieve the position and momentum vectors and the corresponding
 /// 6D covariance matrix (pos:1->3,mom:4-6) of a track at a given z-position
 virtual StatusCode positionAndMomentum( TrTrack* track, double z = 0, ParticleID partId = ParticleID(211),
                                         HepPoint3D pos, HepVector3D mom, HepSymMatrix cov6D );
/// Retrieve the 3D-position vector of a state at a given z-position
 virtual StatusCode position( TrState* state, double z = 0,  ParticleID partId = ParticleID(211), HepPoint3D pos );
```

…

Tools - projectors

- at present these are methods inside the derived TrMeasurement classes
    - in VeloPhiClusterOnTrack, VeloRClusterOnTrack, OTClusterOnTrack, etc.
- proposal to make the projectors as tools
    - no need to load geometry in TrMeasurement derived classes
    - decouples geometry from TrMeasurement derived classes
    - facilitates the converge of the "Measurement" classes for online/offline
- projections should be made in local coordinates
    - done at present in global coordinates,
      i.e. as with perfect geometry/alignment

In short …

Visible to the user:

- tracks
- states
- propagators

To help the tracking/pattern recognition developers:

- nodes and measurements
- projectors

Details on status of implementation:
http://cern.ch/eduardo.rodrigues/lhcb/tracking/event_model/index.html

(note: place of evolving ideas/implementations …)

Plans for next steps

- **implementation of new event model + adaptation of trackfit:**
  - – Jose Hernando + Eduardo Rodrigues
  - ✓ make the TrTrack and TrState base classes available
  - ✓ make the TrMeasurement and TrNode classes available
  - ✓ re-write the extrapolator classes
    - ➢ adapt to new model + introduce new features needed by new model
  - ✓ re-code "user-code" with these new classes
    - ➢ e.g. vertex finding algorithms do not need much more
  - ✓ write the projector tools

A lot can/should be done in parallel

- **start adapting existing algorithms to new event model as soon as header files become available:**
  - – Velo tracking:
    - – combine HLT and offline Velo tracking: Glasgow + Liverpool
  - – VTT tracking: – Yuehong Xie
  - – Forward tracking:
    - – Merge HLT and offline: Olivier Callot & Jose Hernando
  - – Matching: – NIKHEF
  - – KsTracking: – Olivier Callot, Yuehong Xie (?)
  - – Seeding: – Matthew Needham + Gabriel Ybeles Smit

Fast Kalman Fit (Jeroen van Hunen)

- Goals:
  - Understand speed of current fit
  - Possibility to provide fast track state at any point in LHCb
- Current Kalman fit
  - Perform a least squares fit of the measurements with outlier rejection
  - Trace the trajectory along the full B-field map with 5th order Runge-Kutta
  - Include all material walls with the same precision as GEANT
    - Allow for multiple scattering “kinks” in the trajectory
    - Take dE/dx into account
  - Performance
    - Good pulls for long tracks, to be (slightly) improved for others
    - Slow: ~15 msec/track (Pentium III PC)
- Speed (Pent III CPU):

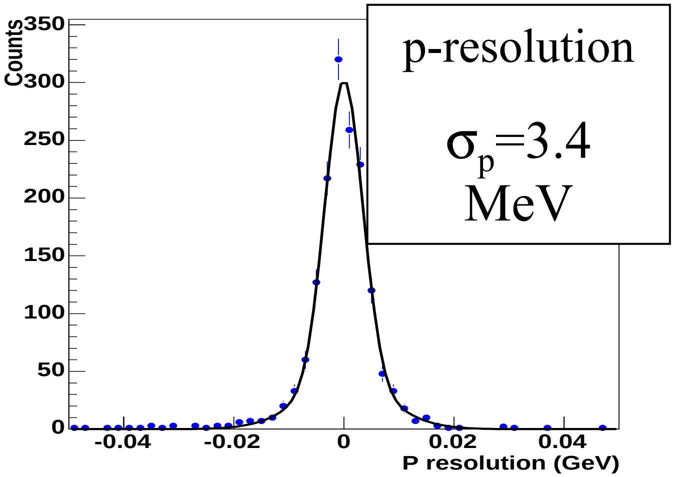
Full fit	15 msec / track
If ignore material walls (skip transport service)	6 msec / track
If replace RK5 by RK4	2.4 msec/track
If tuned B-field service	1.5 msec / track

1.0 msec = B-field access
0.5 msec = Kalman operations

Can be further improved?

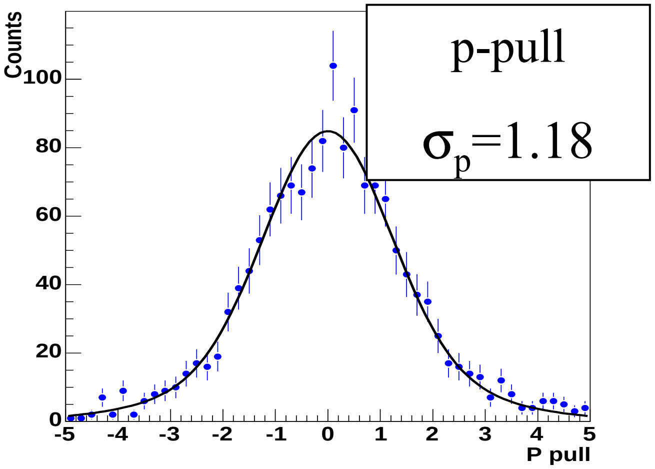
Fast Kalman Fit

15 msec :



p-resolution

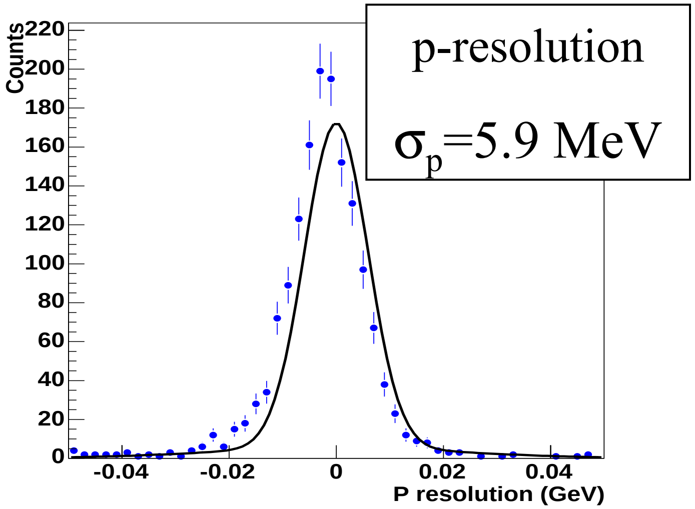
σ_p=3.4 MeV



p-pull

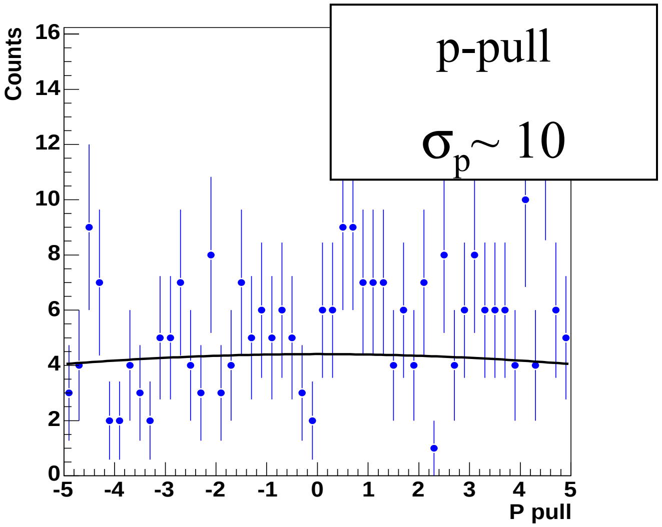
σ_p=1.18

1.5 msec :



p-resolution

σ_p=5.9 MeV



p-pull

σ_p~ 10

Without material the fit provides unusable errors
=> re-introduce fast material walls for fast fit?

Tsa (Matthew Needham)

TSA: Track Seeding Algorithms:

Develop a framework + tools for fast standalone seeding

- Optimize data access in ”DataSvc”, providing iterators over hits according to the geometry structuring
- Provide a set of tools:

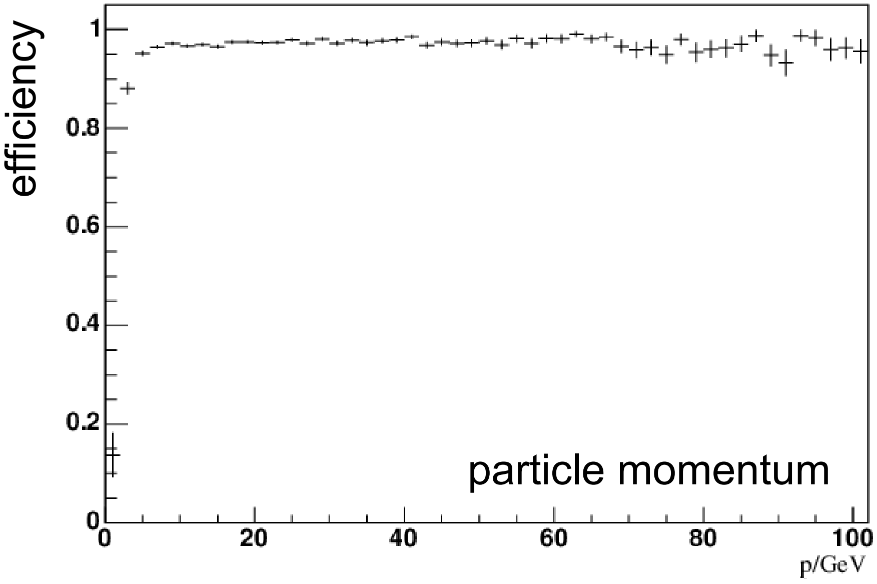
  e.g. “fault calculation”, “track following”, utility classes for parabola’s etc.

- First implementation: IT seeding using spacepoints
  - Create “spacepoints” in IT: ~ 100 spacepoints / event
    - Search for xuvx spacepoints
    - Search for x’u’x spacepoints with unused clusters
    - Search for xuv spacepoints with unused clusters
  - Link the spacepoints
    - Link T1, T2, T3 requiring consistency criteria
    - Calculate a chi2
    - Calculate the number of “faults” (a hit would be expected but is missing)

Tsa – Matthew Needham

Performance for B->J/psi Ks events:

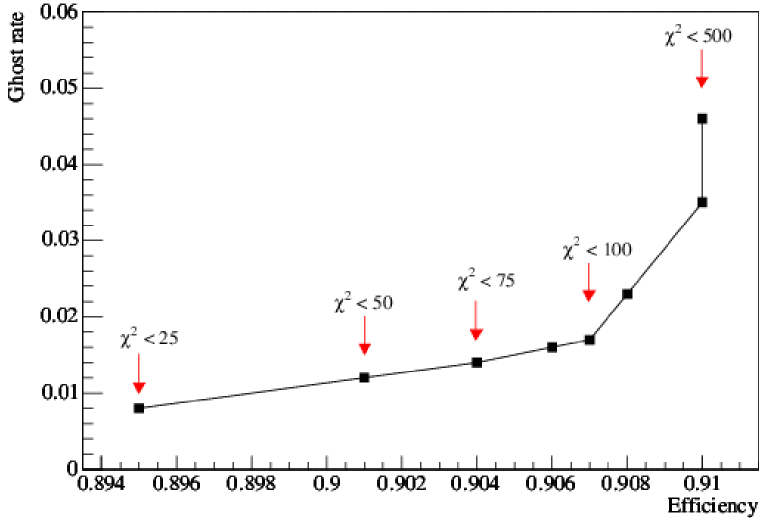
- Efficiency: 97% (B decay tracks)
- Ghost rate: 3.4%
- CPU time: 7.9 ms per event

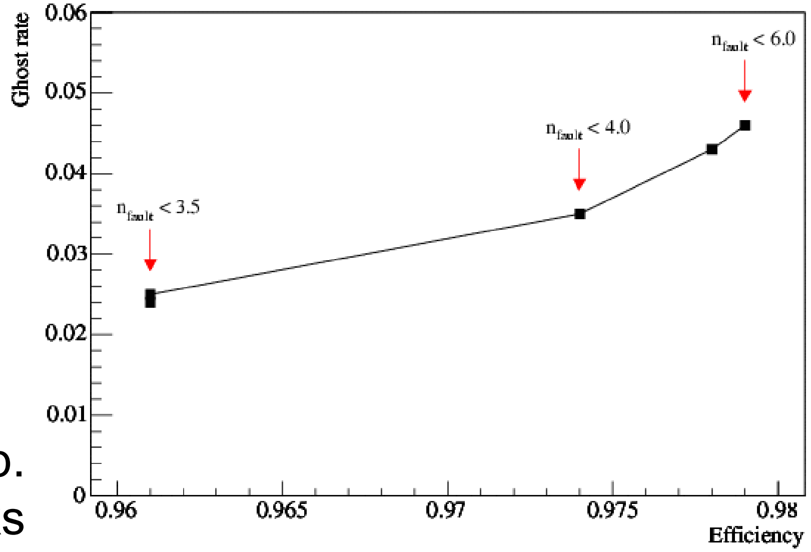


OT version is underway.

More difficult due to the presence of:

- Hot spots area’s with more than 30% occup.
- Chains of consecutive hits from steep tracks





Summary

Job-list:

- Still room for improvement in reconstruction algorithms
  - Help is welcome
- Online and offline reconstruction are being combined in new track event model
  - Many changes in the track fit
  - Adapting the pattern recognitions
- As the subdetectors are starting to deliver “misaligned data” the reconstruction must be prepared to deal with it
  - Still to be started: help is needed.
- New ideas and algorithms are welcome (Tsa, fast fit, …)