RAVE for the ILC community
- first results

Wolfgang Waltenberger, Bernhard Pflugfelder
Institute for High Energy Physics of the Austrian Academy of Sciences

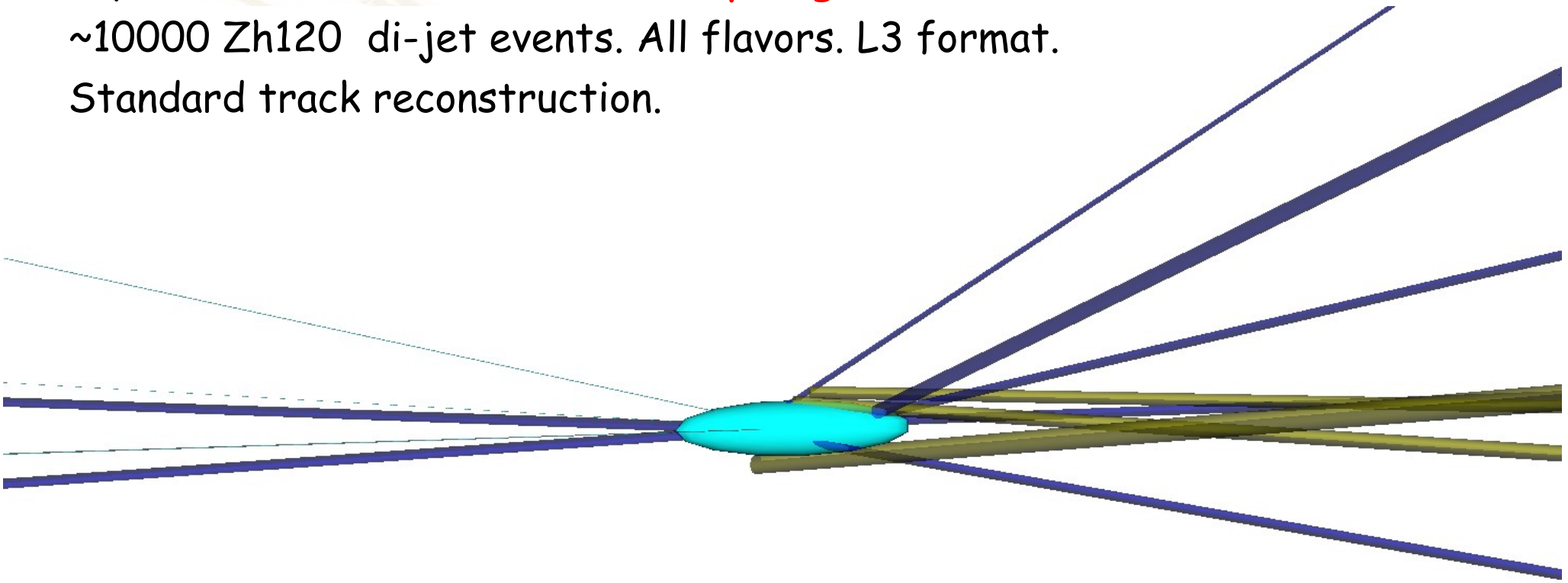
Hamburg, May/June 2007

Input data

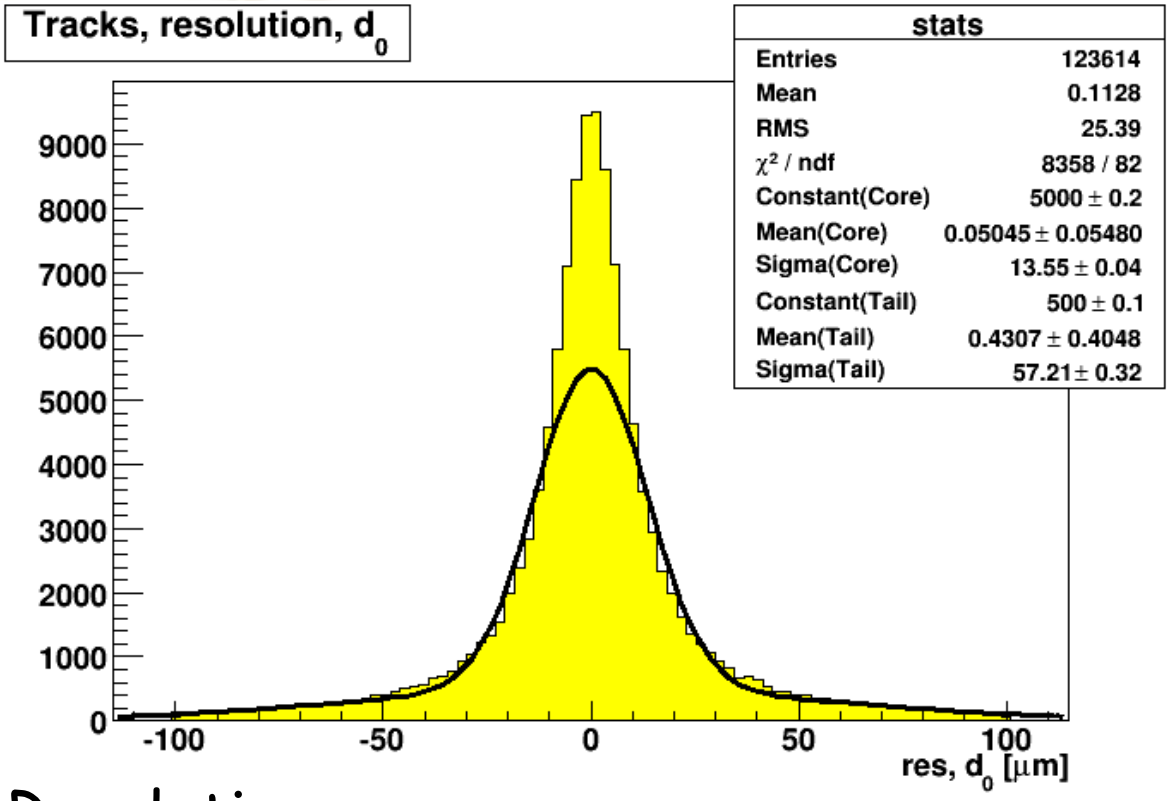
Input data are due to Ben Jeffrey, big thanks!

~10000 Zh120 di-jet events. All flavors. L3 format.

Standard track reconstruction.



Track parameters: d_0



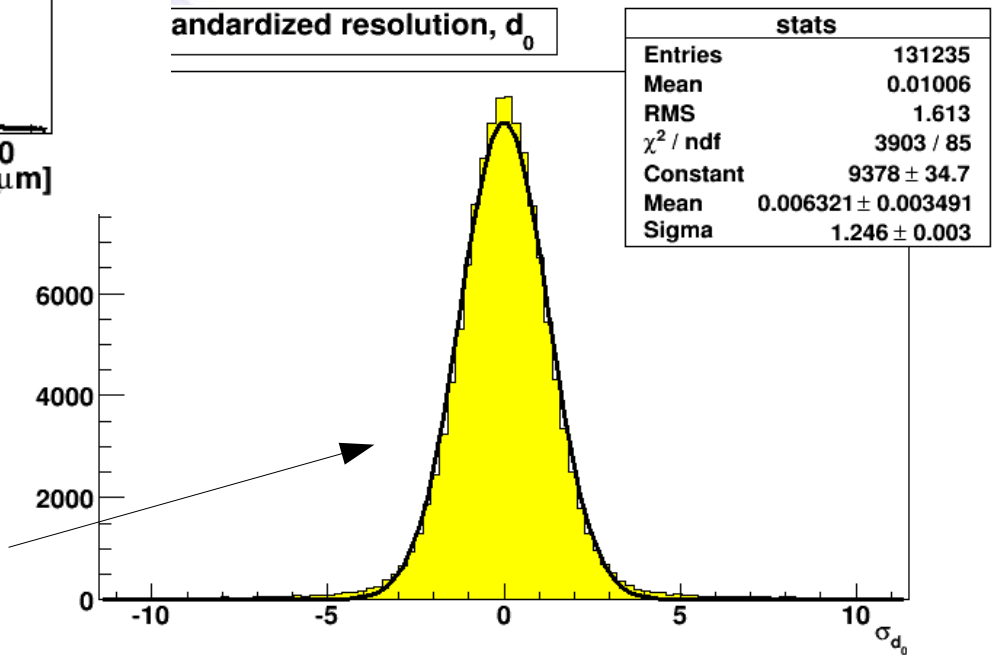
Converted to "euclidean" Rave coordinates, then to "curvilinear" coordinates! Good check for conversion!

Resolutions:

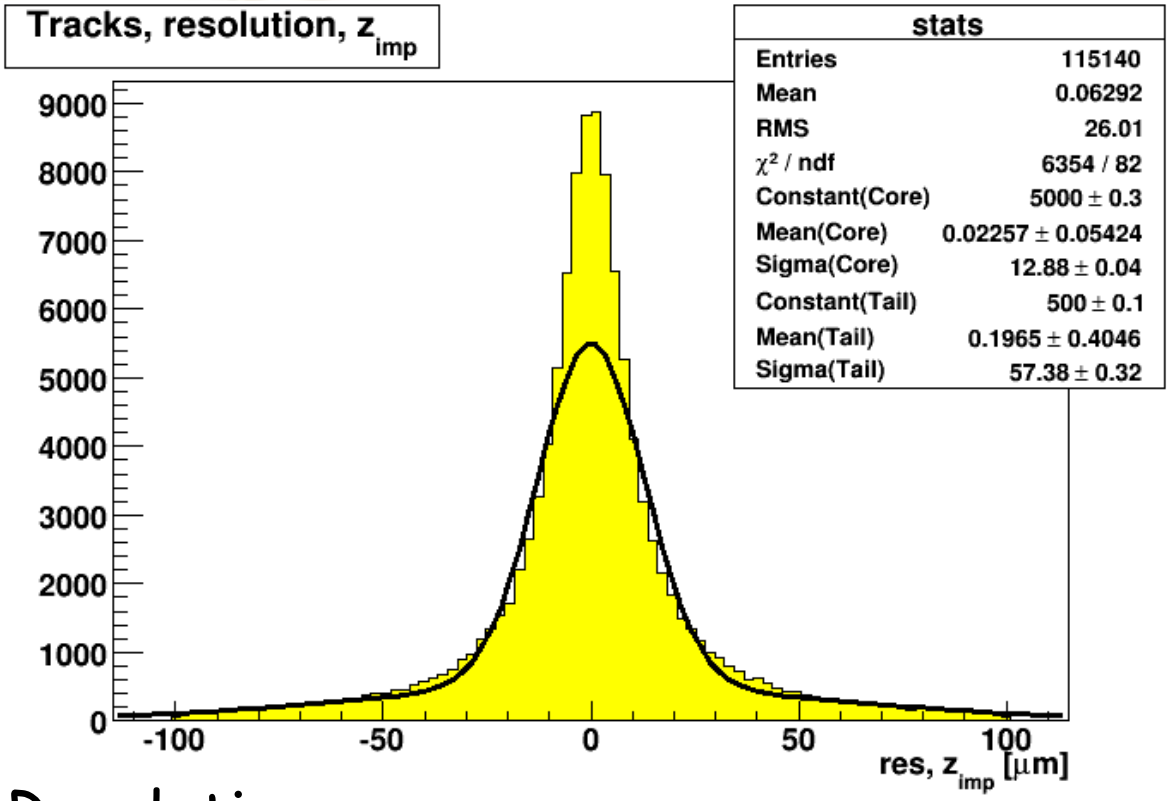
σ ~ 10 μm ("core")

σ ~ 60 μm ("tail")

"pulls": N(0.006, 1.25) good!



Track parameters: z_0

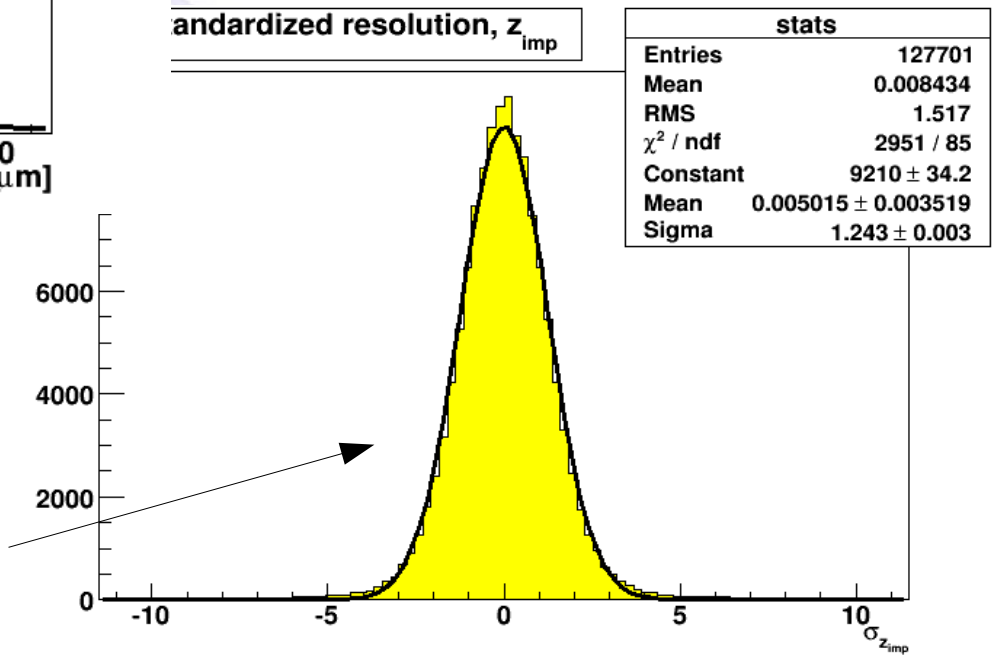


Same picture!

Resolutions:

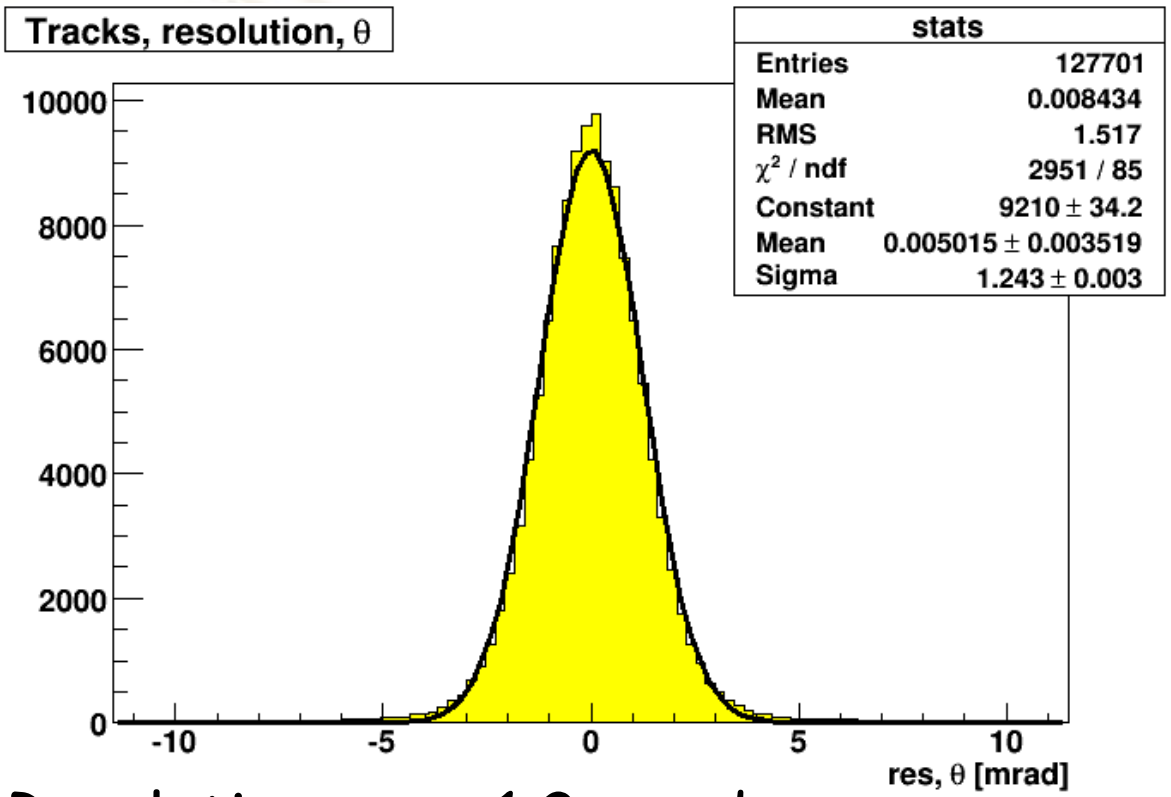
σ ~ 10 μm ("core")

σ ~ 60 μm ("tail")



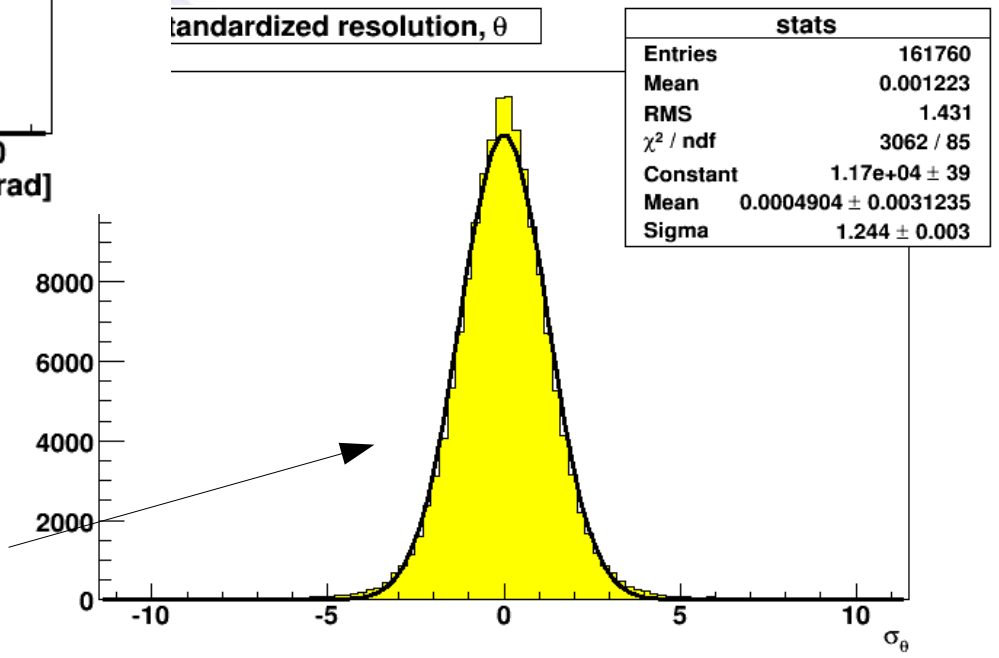
"pulls": N(0.005, 1.24) good!

Track parameters: θ



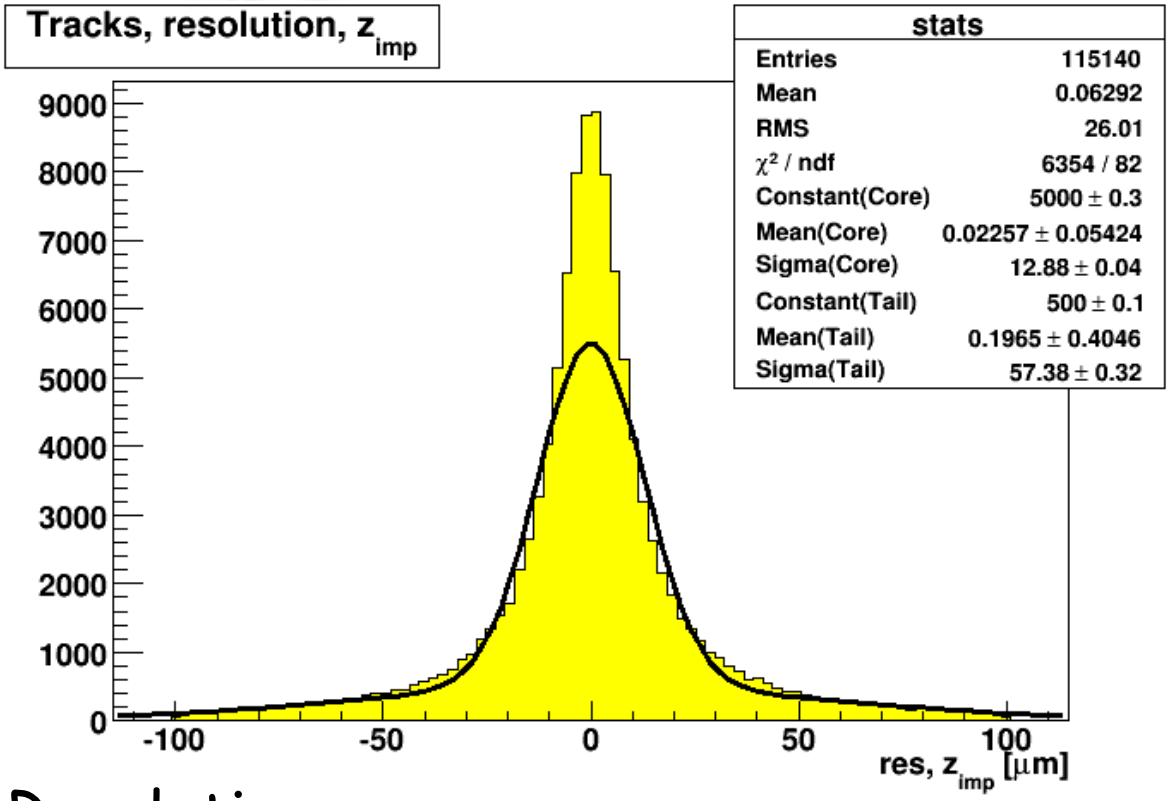
FIXME definition of theta ... tan lambda?

Resolutions: σ ~ 1.2 mrad



"pulls": N(0.004, 1.24) good!

Track parameters: z_0

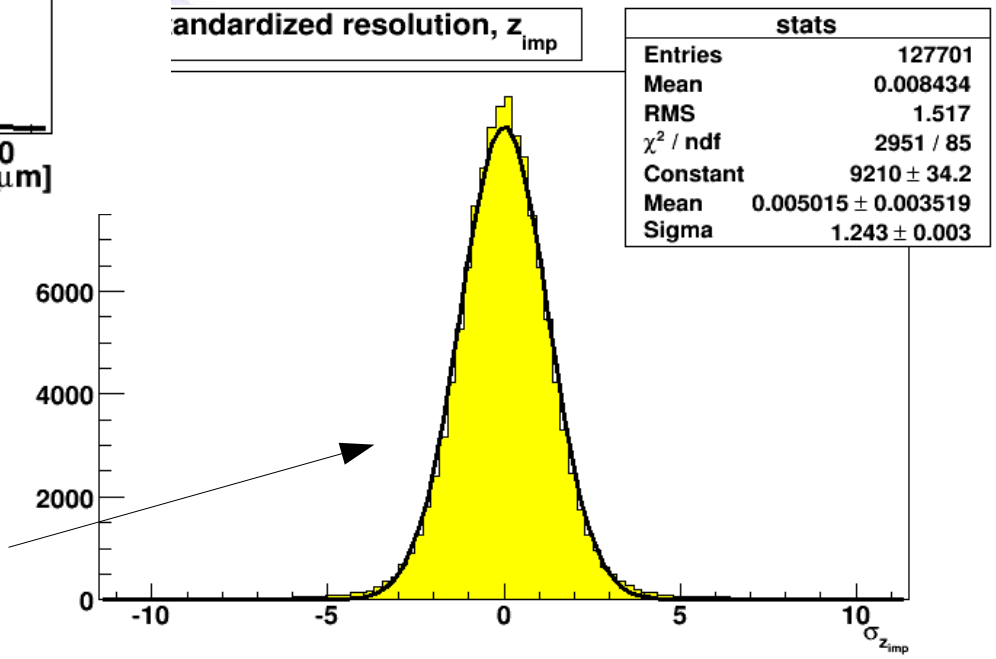


Same picture!

Resolutions:

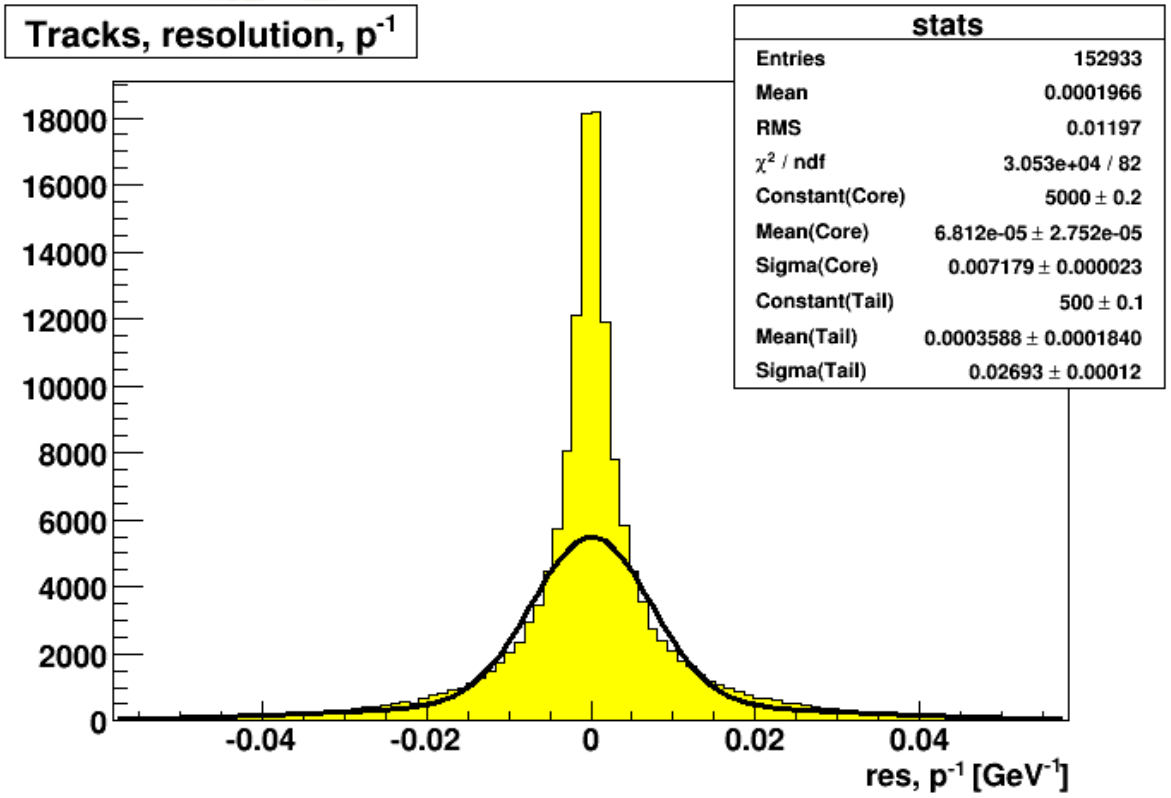
σ ~ 10 μm ("core")

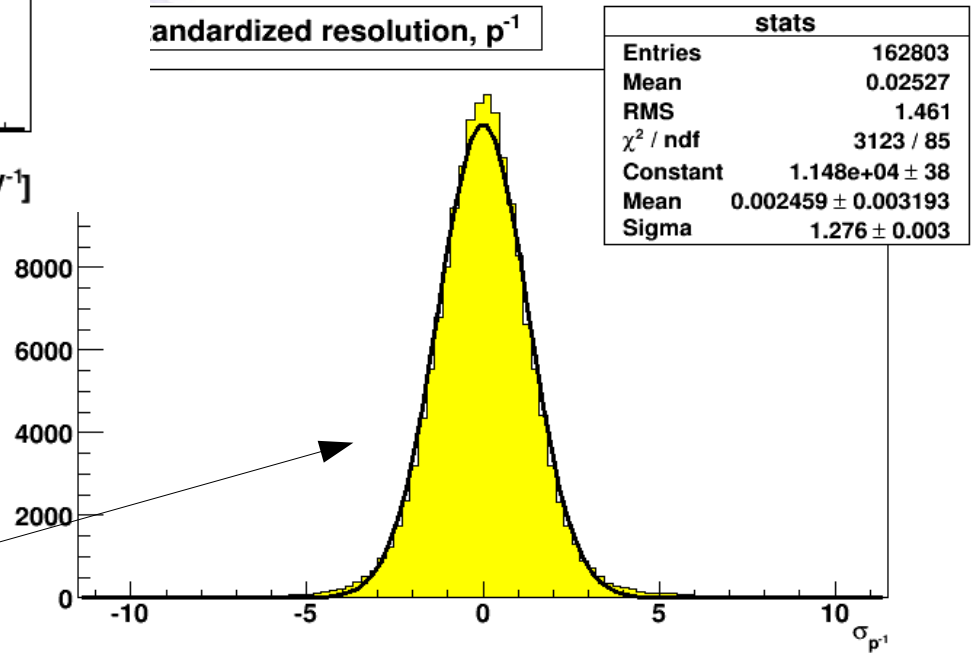
σ ~ 60 μm ("tail")



"pulls": N(0.005, 1.24) good!

Track parameters: p^{-1}





"pulls": N(0.002, 1.276)

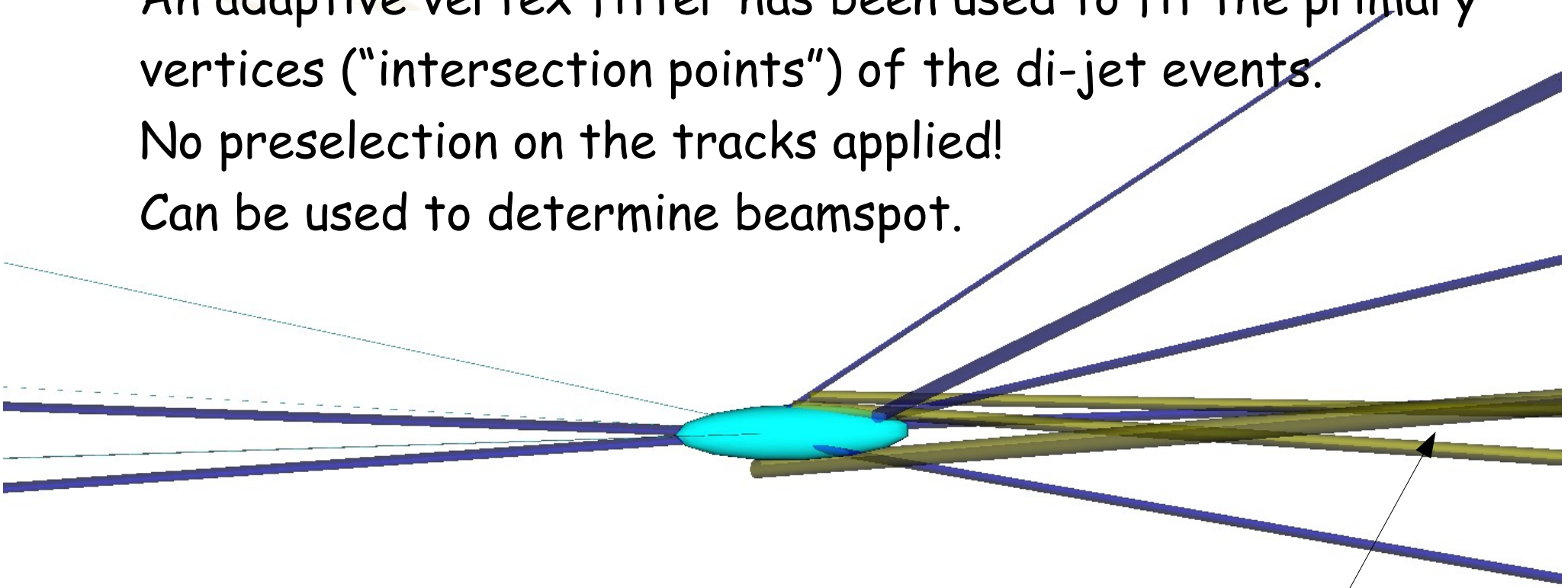
Summary input data

All track parameter errors seem to be underestimated but acceptable.

It is considered an asset of the adaptive (vertex) fitting methods to be able to deal with imperfect data.

Fitting the intersection point

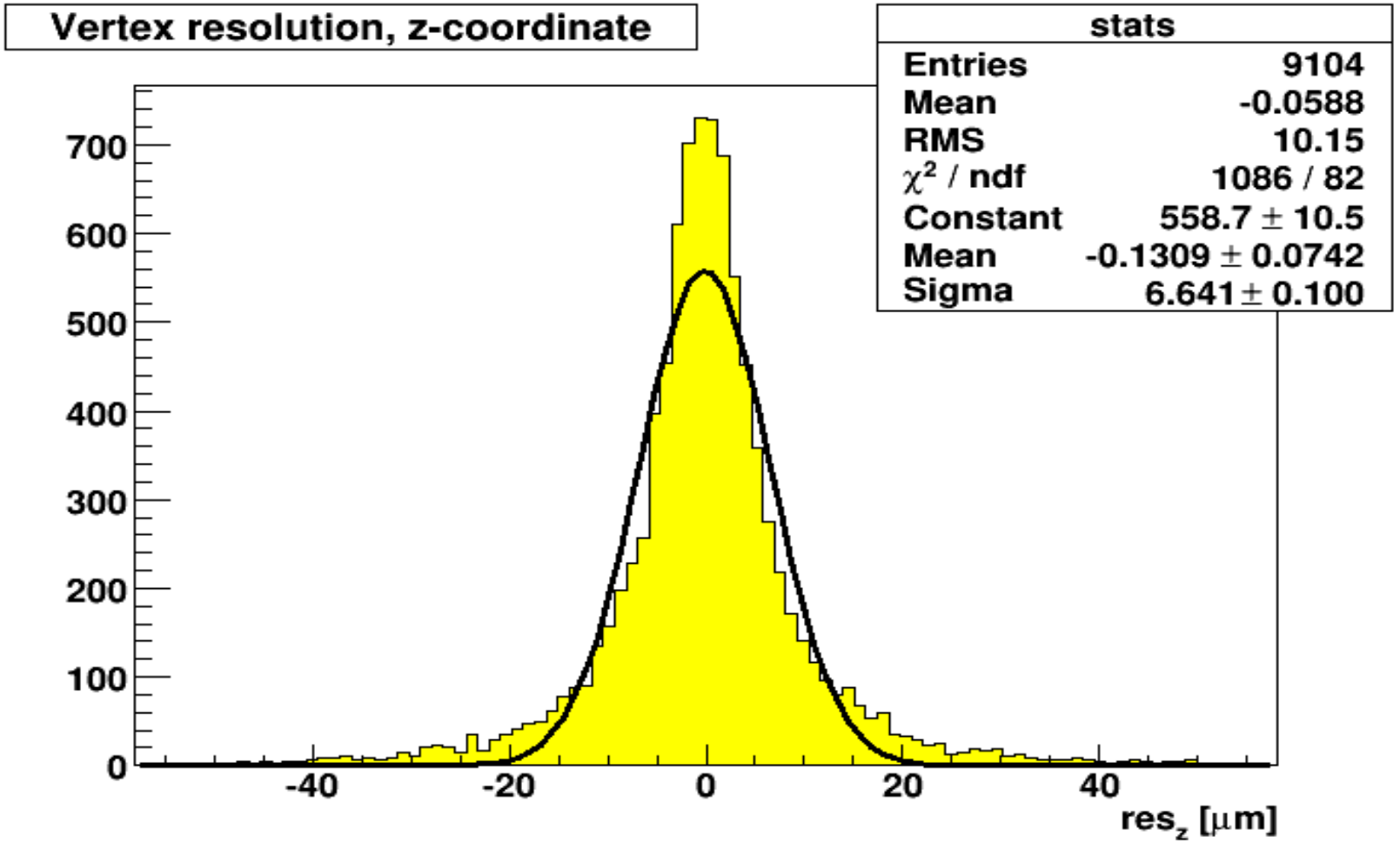
An adaptive vertex fitter has been used to fit the primary vertices ("intersection points") of the di-jet events.
No preselection on the tracks applied!
Can be used to determine beamspot.



Fitted primary vertex, magnified x20.
(A charmed event, and yet so many primary tracks?)

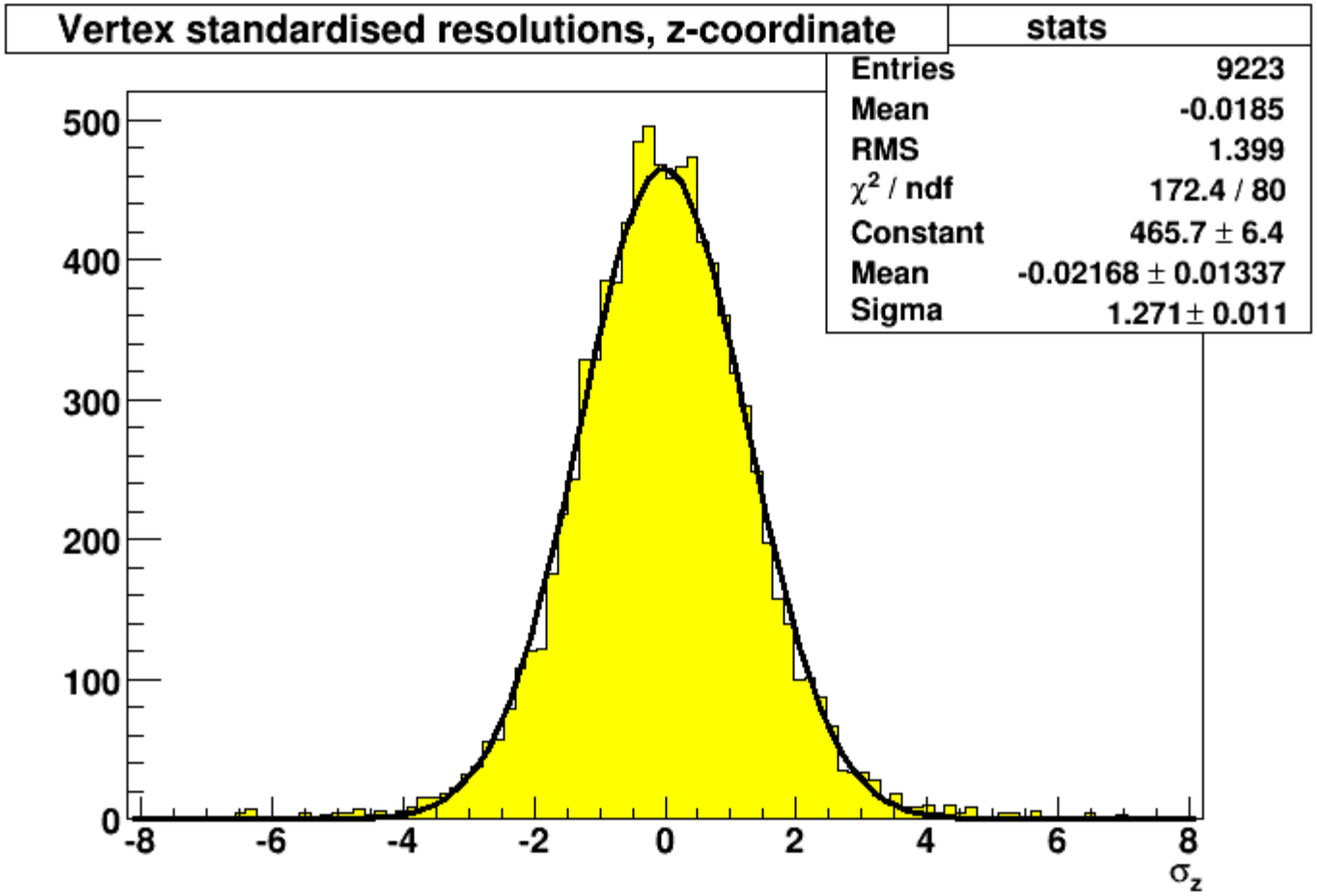
D^0

Fitting the intersection point



Resolution in z: 6~7 μm (if we ignore the b- and c-jettish events)

Fitting the intersection point

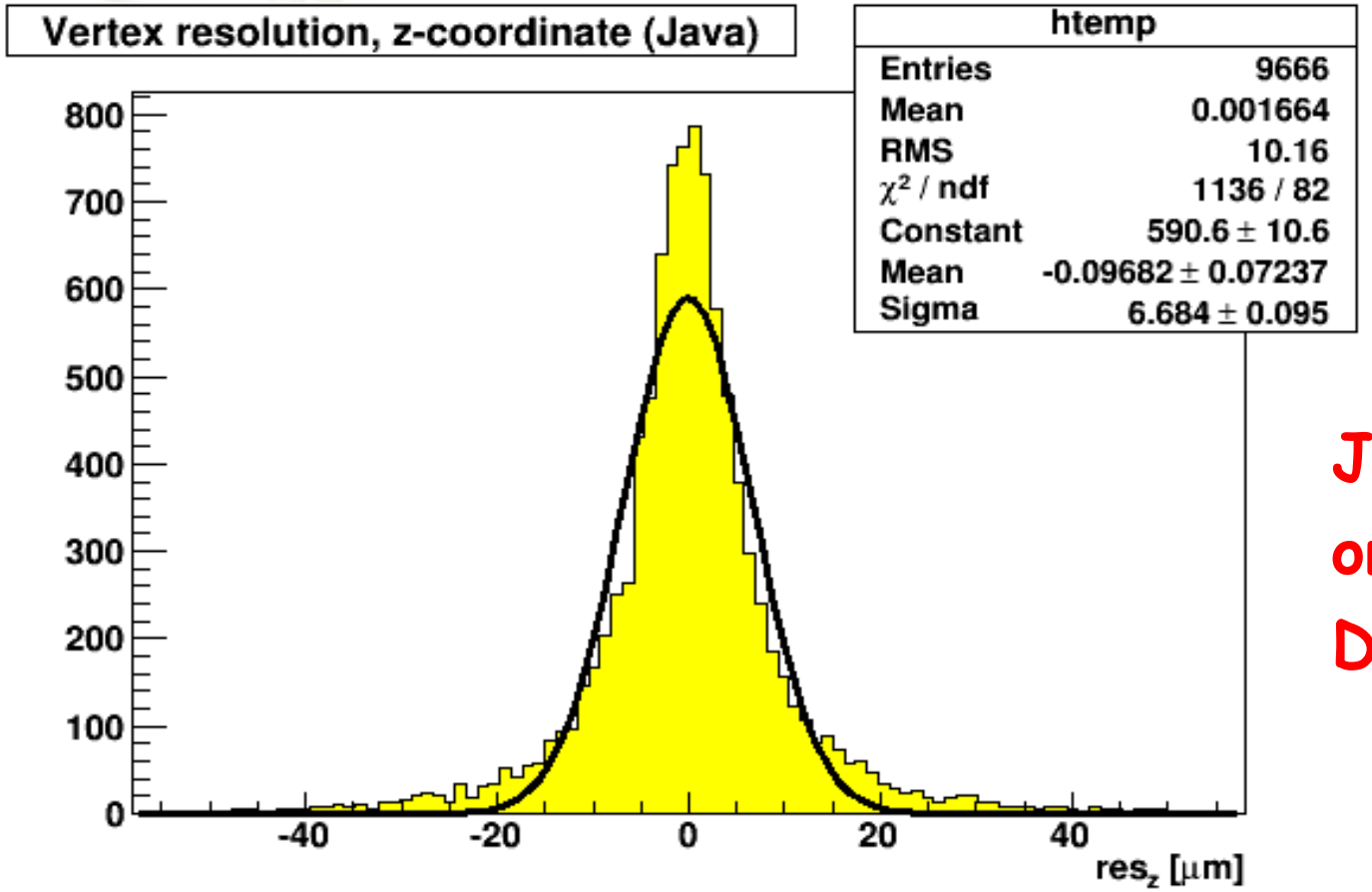


"Pulls" of z-coordinate: N(-0.02, 1.27).

Very similar to tracks' pulls.

Java and Cygwin

Rave now also runs as a Java library, and under cygwin.



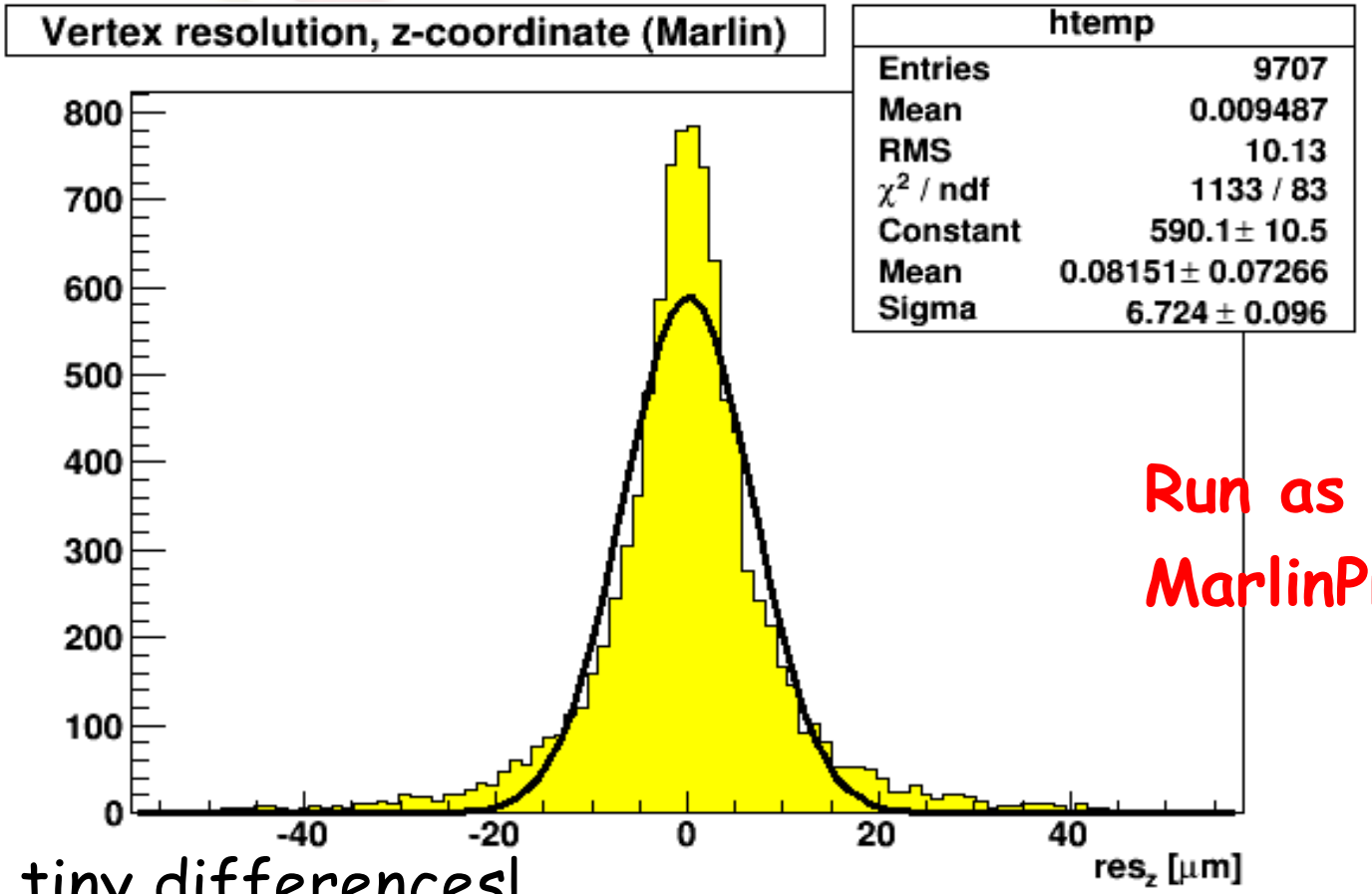
JavaRave!
org.lcsim-Driver!

Same events, same code, but run as an org.lcsim-Driver - slightly (but **only** slightly) different results?!

Java and Cygwin

Rave now also runs as a Java library.

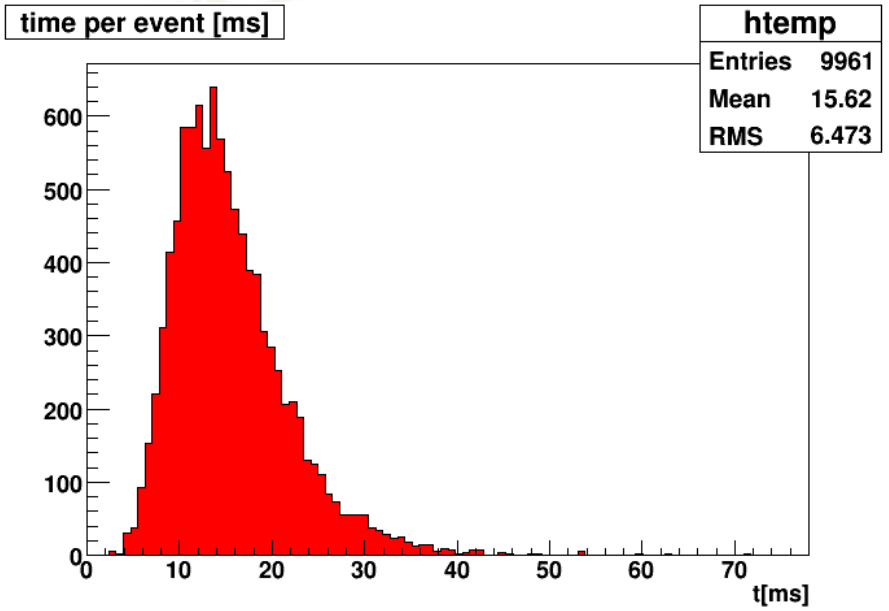
It also compiles under cygwin.



Run as a MarlinProcessor!

Again, tiny differences!

Fitting the intersection point

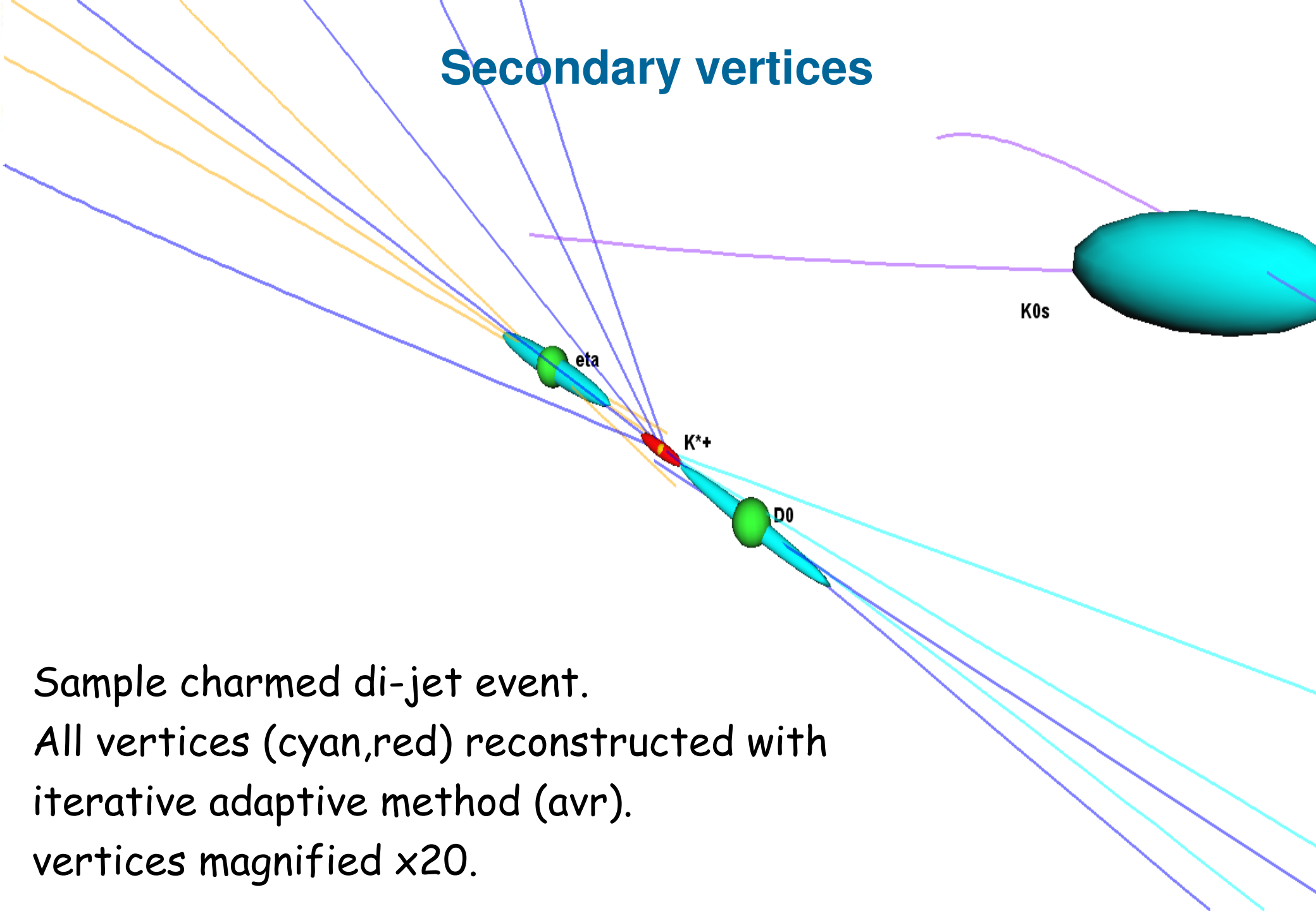


	resolution [μm]	pulls, bias	pulls, σ
x	7	0.04	1.3
y	7	0.01	1.3
z	7	-0.01	1.3

Primary vertices can be reconstructed without prior track selection, without extra information, with ≈15ms per event (Intel Xeon 3GHz, 512kb Cache)

(should be faster – overhead due to conversions? L2 Cache?)

Secondary vertices

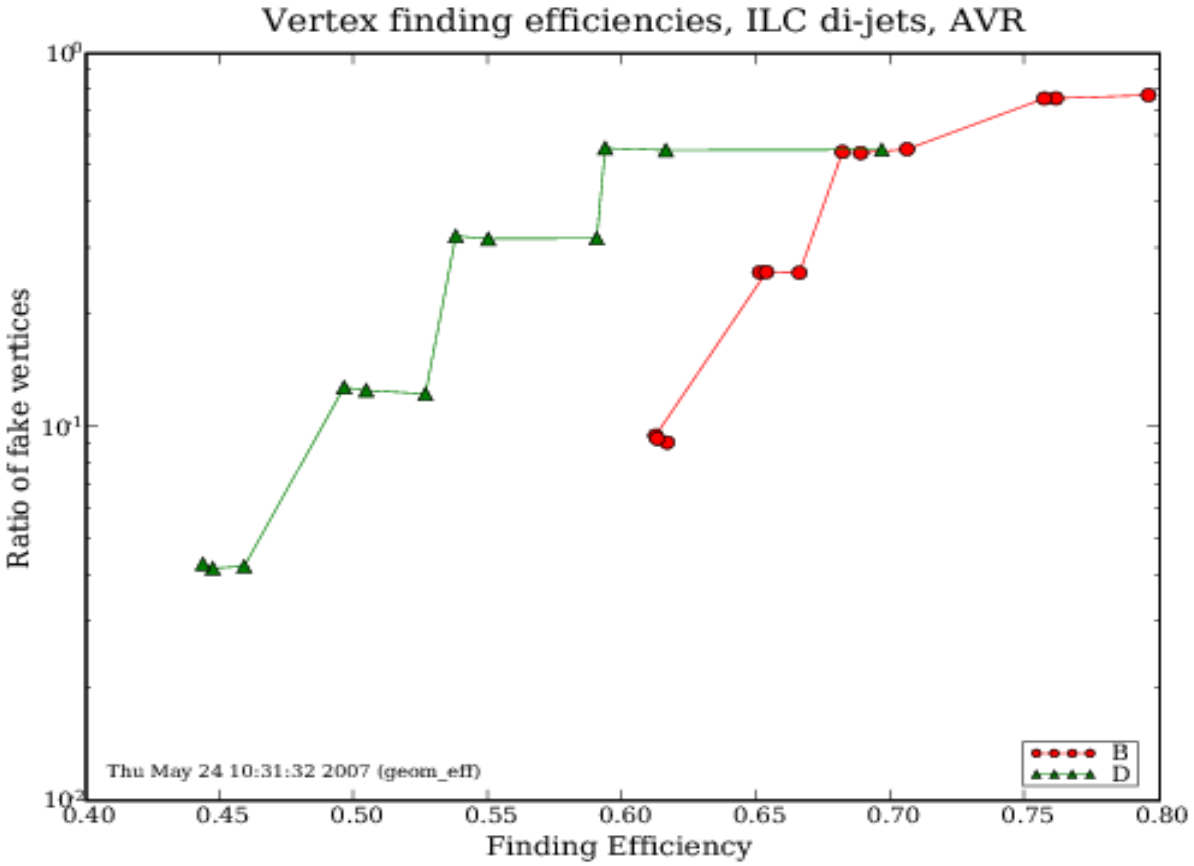


Sample charmed di-jet event.
All vertices (cyan,red) reconstructed with iterative adaptive method (avr).
vertices magnified x20.

Secondary vertex finding

First attempts have been made to run an iterative adaptive vertex reconstructor (avr) on the dijets to find and fit the primary vertex as well as all secondary vertices.
Algorithm has been run on the events, not on the individual jets! Again, no prior selection, algorithm has been run "out of the box".
Reconstructed vertices have been associated with simulated vertex, if > 50 % of the tracks are "in common".

Vertex finding efficiencies



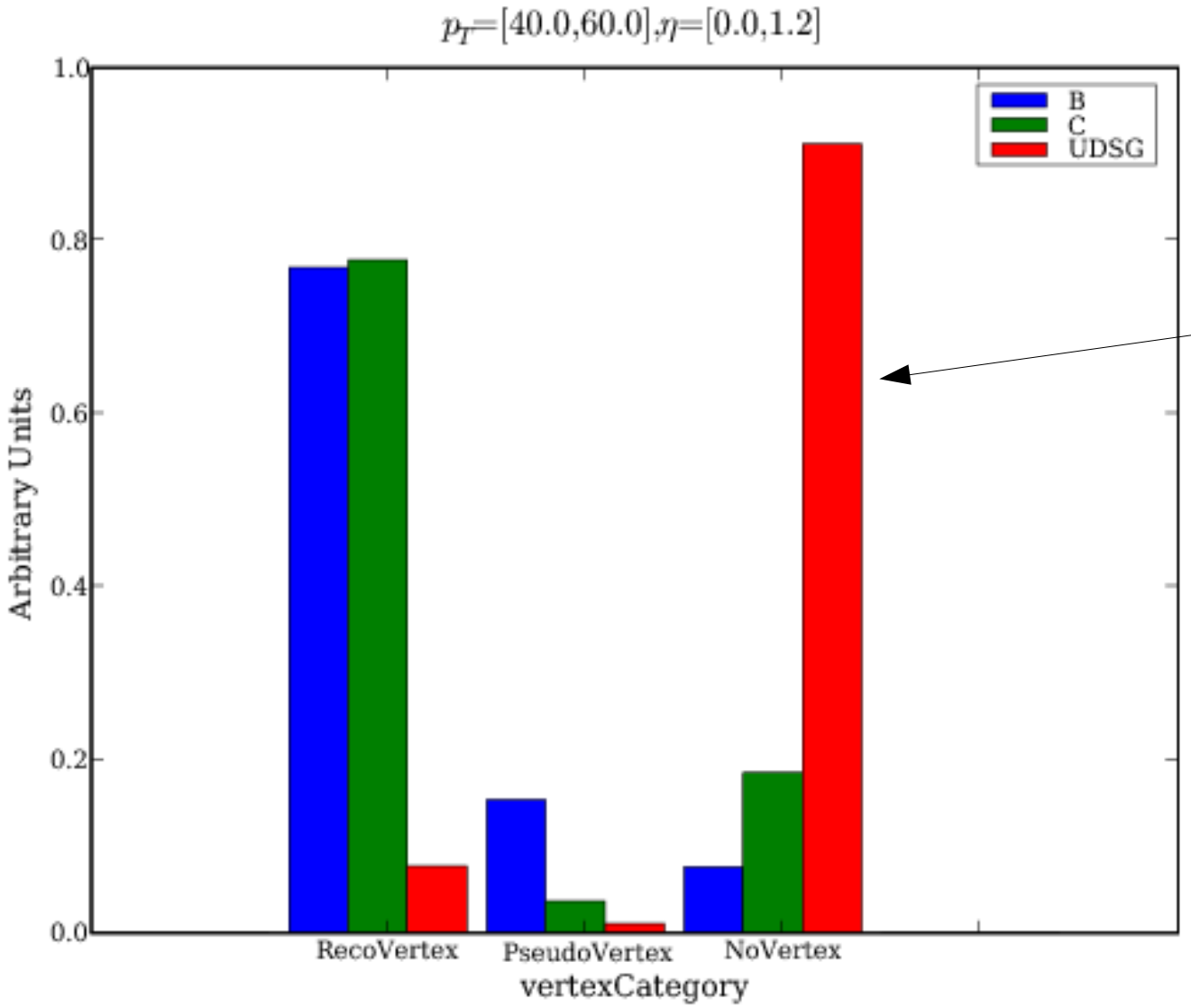
Absolute numbers not very meaningful – can only be used to qualitatively compare algorithms.

B-Tagging

- The standard Rave b-tagger has been tried.
Simple likelihood-ratio method, code for B **and** D tagging exists, but has not been used yet.
- Jet flavor: TrueAngularJetFlavourProcessor
- Tagging variables are track-, vertex-, and jet-based.
"Vanilla" AdaptiveVertexReconstructor used for vertex finding/fitting.
- Training sample: Ben Jeffrey's 10000 events. (Way too small for calibration).
- Fancier methods will follow: neural nets, boosted decision trees, genetic algorithms, etc.

Tagging Variables

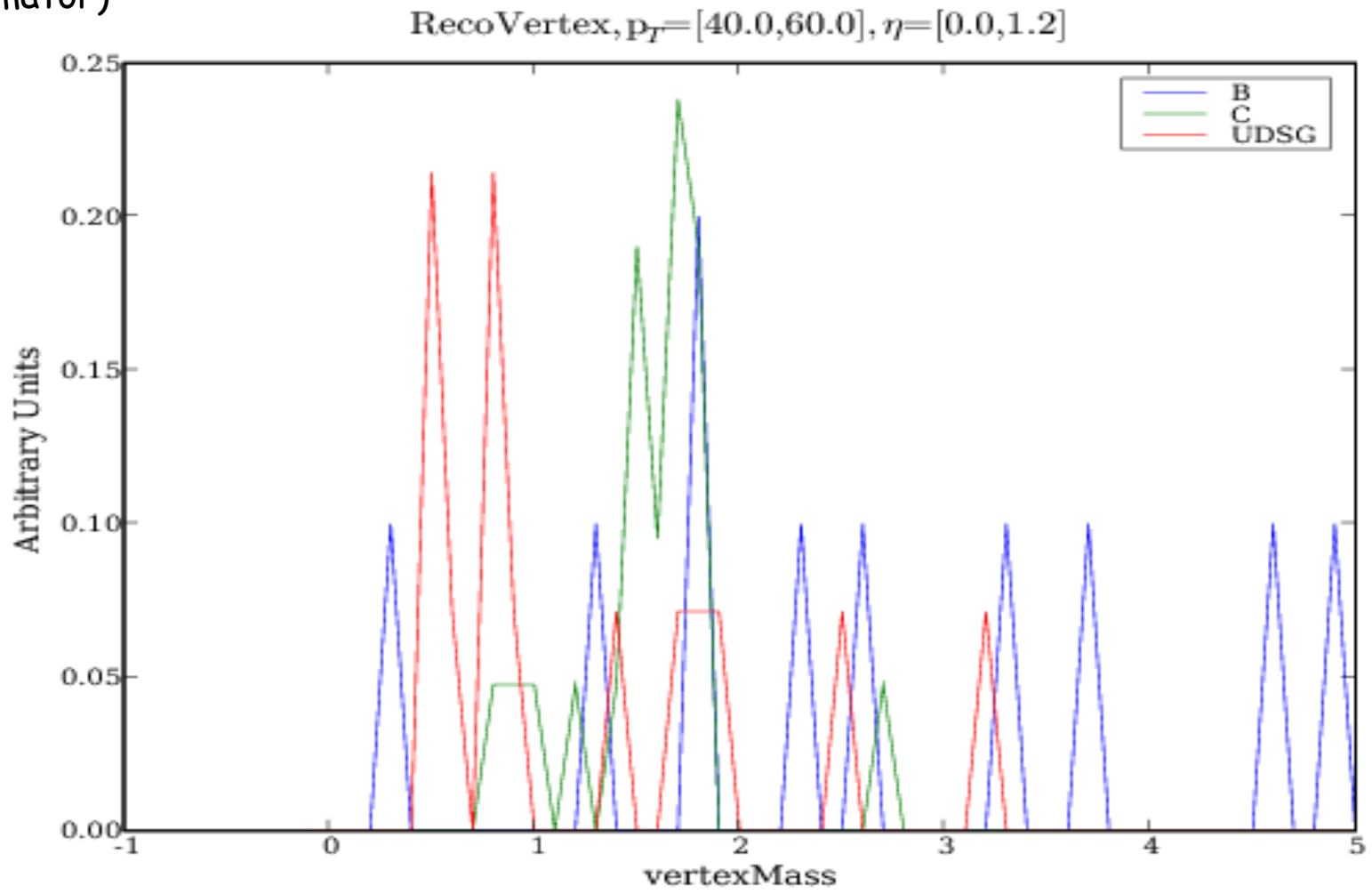
PDFs of some tagging variables:

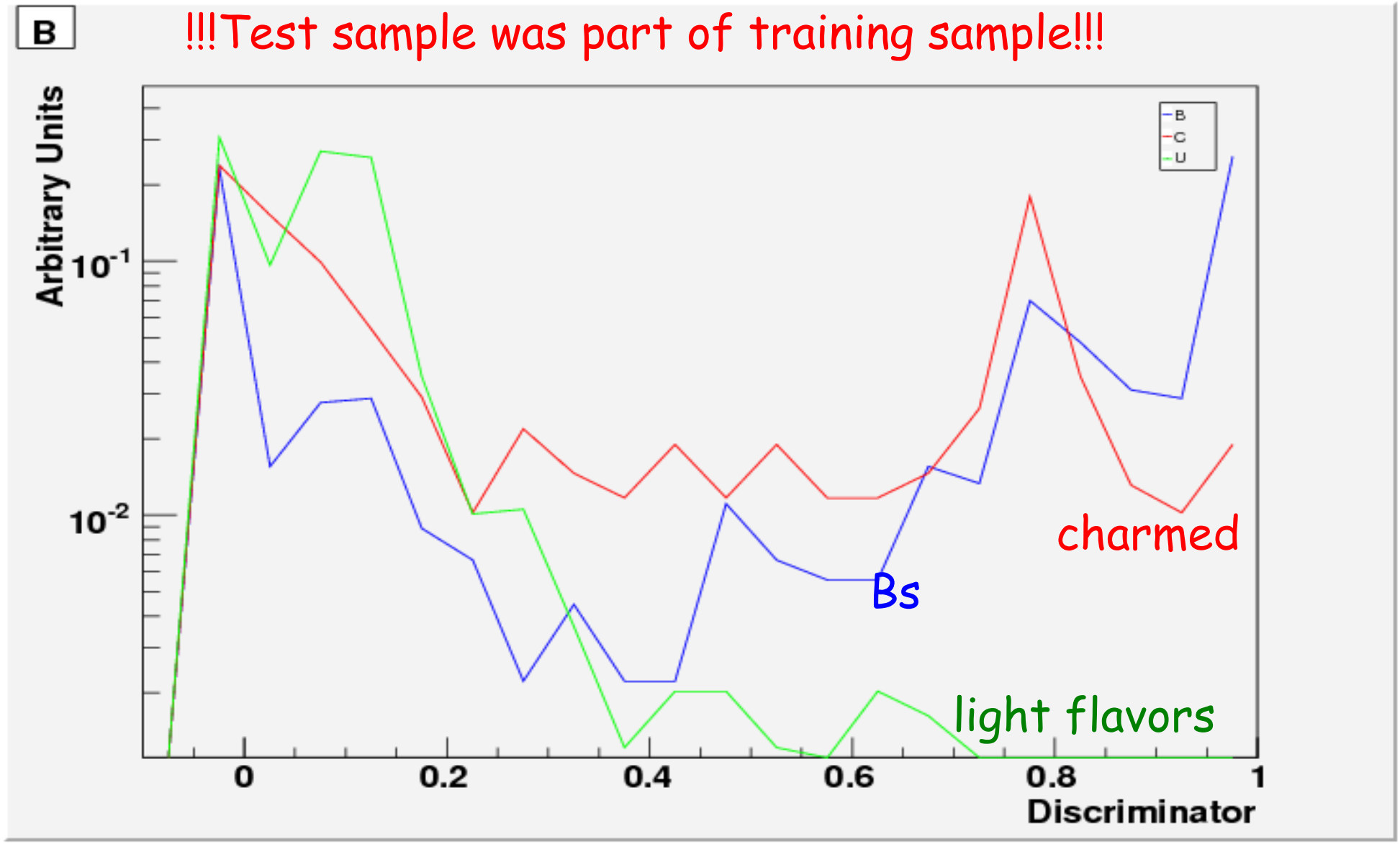


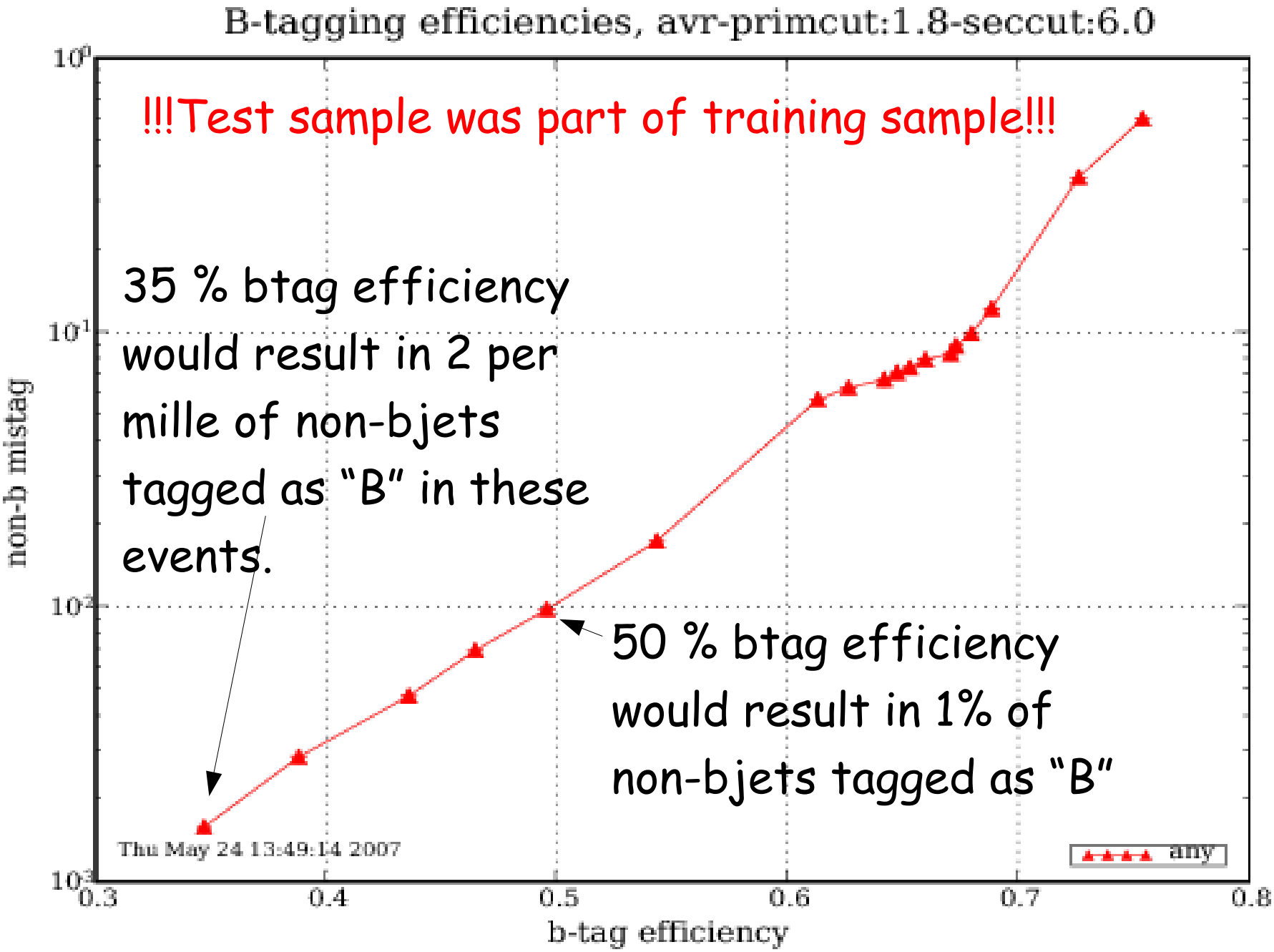
"No vertex found" means it's probably a light jet

Tagging Variables

"Mass at vertex" - too few events! (Will try with kernel estimator)




B
!!!Test sample was part of training sample!!!
Arbitrary Units
10^{-1}
10^{-2}
B
C
U
charmed
Bs
light flavors
0
0.2
0.4
0.6
0.8
1
Discriminator


B-tagging efficiencies, avr-primcut:1.8-seccut:6.0
!!!Test sample was part of training sample!!!
35 % btag efficiency would result in 2 per mille of non-bjets tagged as "B" in these events.
50 % btag efficiency would result in 1% of non-bjets tagged as "B"
non-b mistag
b-tag efficiency
Thu May 24 13:49:14 2007
any

Caveat

B-Tagging results should not yet be taken too seriously:

- too small, too special training sample
- test sample was part of training sample!!!
- algorithms used out-of-the-box

etc.

Will need big event samples (where from?)

Conclusion

Rave has successfully been tried on ILC data for vertex finding, vertex fitting, and b-tagging.

Very decent, acceptable first results.

But of course, many, many things can still be improved.

Future developments:

- Turn B-Tagger into Flavor-Tagger
- Compare iterative AdaptiveVertexReconstructor with other methods (Zvres etc.) -> code exists already!
- Try fancier learning algorithms -> code exists already!
- Marlin processor exists, but currently exposes only vertexing capabilities. Org.lcsim Driver exists, also.

References

- [CMS Note 2007/008] "Adaptive Vertex Fitting"
http://cmsdoc.cern.ch/doc/notes/docs/NOTE2007_008

- "Adaptive Multi-Vertex Fitting", CMS CR-2004/062,
http://cmsdoc.cern.ch/documents/04/cr04_062.pdf, CHEP proceedings
Interlaken

- RAVE: http://projects.hepforge.org/rave/trac/wiki

- MarlinRave: Glue code for Marlin
http://stop.itp.tuwien.ac.at/websvn/listing.php?repname=marlinrave

- org.lcsim Rave: Glue code for org.lcsim
http://stop.itp.tuwien.ac.at/websvn/listing.php?repname=lcsimrave