Fast simulation and topological vertex finding in Java

Wolfgang Walkowiak,
UC Santa Cruz

Linear Collider Workshop 2000,
Fermilab,
26 Oct 2000

Overview

- Fast Monte Carlo simulation in LCD Java Analysis Studio framework:
  - Track smearing
  - Calorimeter cluster smearing
- Implementation of topological vertex finding algorithm

Fast MC in LCD Java/JAS

- Fast Monte Carlo in LCD Java/JAS environment exists:
  - Tracks are generated from MC particle information:
  - Five helical track parameters from MC particles:

    d_0 – distance in xy plane from orbit to origin

    φ_0 – azimuthal angle in xy plane (track direction)

    ω – signed geometrical curvature

    z_0 – z position of orbit at doca point

    $\tan\lambda$ – tangent of the dip angle

  - New: smearing of these parameters with full covariance matrix.

Covariance matrices

- Track smearing uses full covariance matrices calculated with program lcdtrk by Bruce Schumm
  - http://www.slac.stanford.edu/~schumm/lcdtrk20000928.tar.gz
  - Lcdtrk uses the Billoir method to calculate the covariance matrix for the 5 track parameters as a function of momentum p and dip angle tan(λ).
  - Covariance matricies for S2 and L2 detector designs are currently available.
  - Tool lcdtrk is easy to use.
- *Note: Recently, a fundamental bug in the covariance matrix calculation for the barrel-endcap transition region was fixed (Sept. 2000). The bug had probably no consequence on generated covariance matrices.*

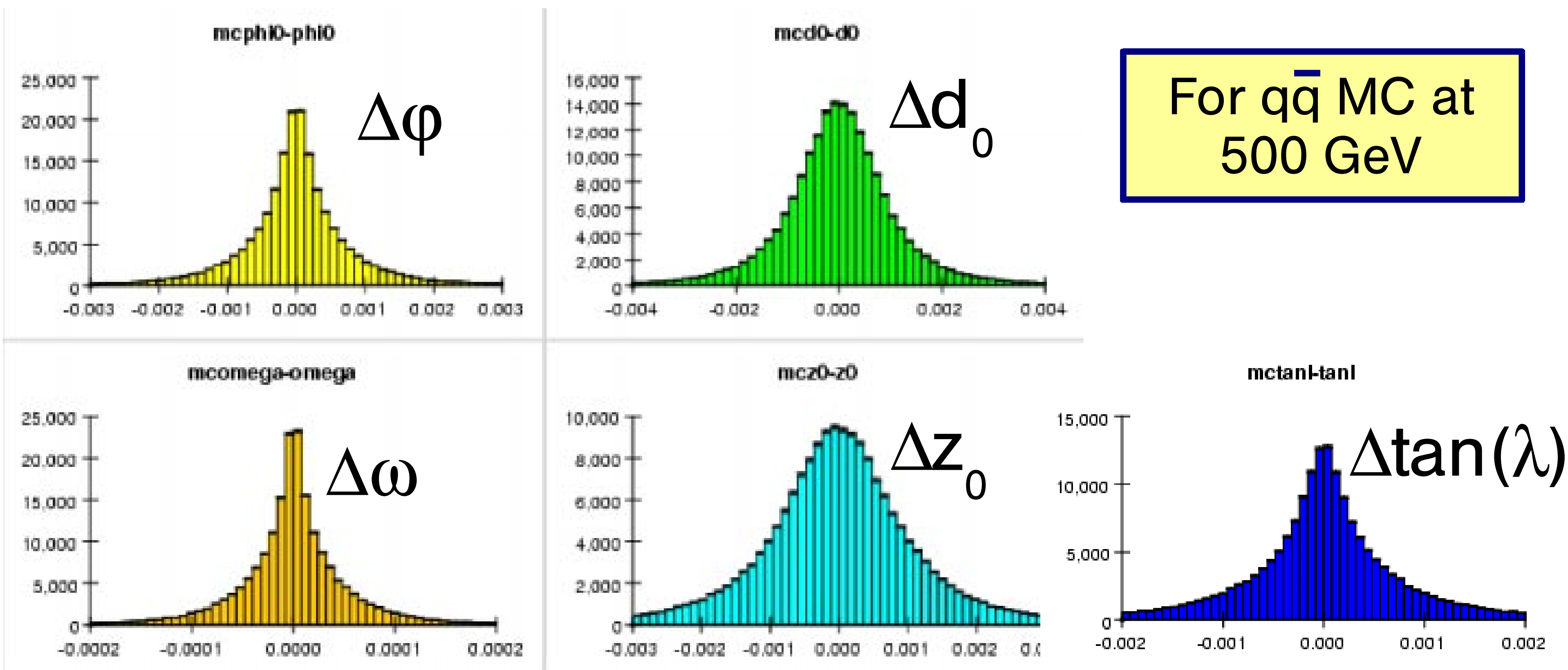
Covariance matrices -- input

```
0 Following the tag line, the information contained in each succeeding
0 line is as follows:
0   1    -> Number of total momentum points (20 maximum)
0   2    -> Value of total momentum at each point (GeV/c)
0   3    -> Number of cos(0) points
0   4    -> Value of cos(0) at each point
0 5 to
0 5+N-1  -> Parameters for each of the N layers (see below for more
0           information on how one specifies a layer
0  5+N   -> The value '-999.' to specify that this is the last line
0           in the file, followed by the B field in Tesla.
....
0 1234567
  13
  1.0 1.8 3.2 5.6 10. 18. 32. 56. 100. 180. 320. 560. 1000.
  44
 .000 .109 .206 .292 .369 .438 .499 .553 .602 .645 .684 .718 .749 .776 .800 .822 .842 .859 .874 .888 .900 .911 .921 .929
    .937 .944 .950 .955 .960 .965 .968 .972 .975 .978 .980 .982 .984 .986 .987 .989 .990 .991 .992 .993
   1    .00    .00     2.0    .000  -9999.00  9999.00  -1   Beam contraint
   1   1.00    .14  99999.0    .000    -3.50     3.50   1   1/2mm of Be
   1   1.19    .06     5.0    .000    -2.50     2.50   1   VXD Layer One must
   1   1.21    .06     5.0   1.571    -2.50     2.50   1   must go here
   1   1.05    .12  99999.0    .000   -3.675    -3.55   1   __
   1   1.05    .12  99999.0    .000     3.55    3.675  1   |
   1   1.15    .12  99999.0    .000   -3.888    -3.65   1   | This is the slanted section of
   1   1.15    .12  99999.0    .000     3.65    3.888  1   | the beampipe. The true thickness
  ....
```

Momentum and tan(λ) points for evaluation.

Program allows for different geometries specified by user!

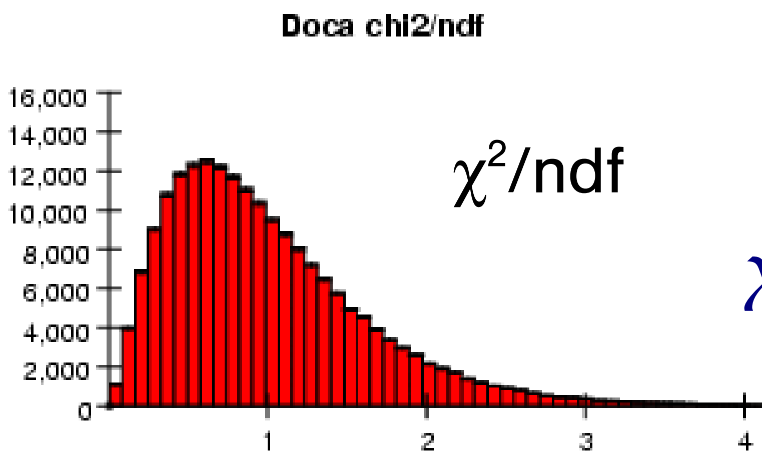
Fast MC track smearing

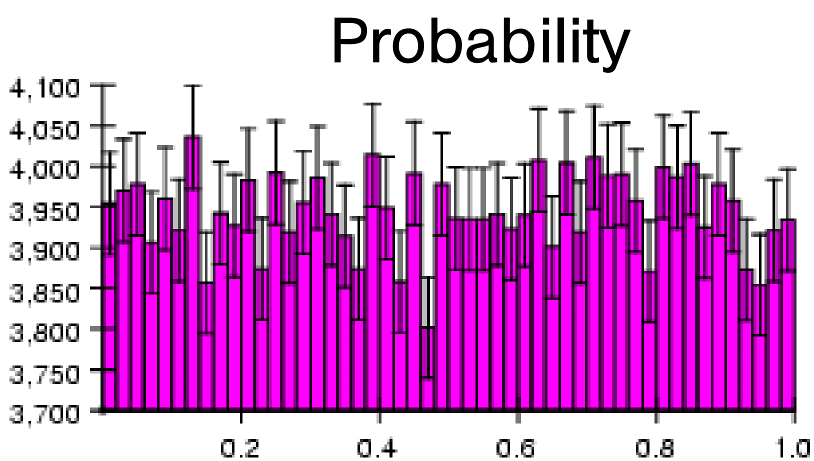


For $q\bar{q}$ MC at 500 GeV

Units in cm

Fast MC track smearing





For $q\bar{q}$ MC at 500 GeV

$$\chi^2 = \sum_i \sum_j \left(a_i^{true} - a_i\right) M_{ij}^{-1} \left(a_j^{true} - a_j\right)$$

- ✔ χ^2/ndf distribution is ok
- ✔ Probabilty distribution is ok.

Fast MC -- calorimetry

- JAS framework allows for an easy implementation of a fast calorimetry simulation.
- Preliminary versions (e.g. by M. Ronan) exist.
- Cluster smearing exists in ROOT environment.
- Development and integration into LCD JAS framework is driven by user demand and input.

Topological vertex finding

- SLD collaboration has developed a unique topological vertexing algorithm, named ZVTOP, which is used by many other collaborations as well.
  (D.J.Jackson, NIM A388 247-253,1997)
- Implementations for LCD studies:
  - ROOT implementation by T. Abe
    (See vertexing session.)
  - JAVA implementation in JAS framework

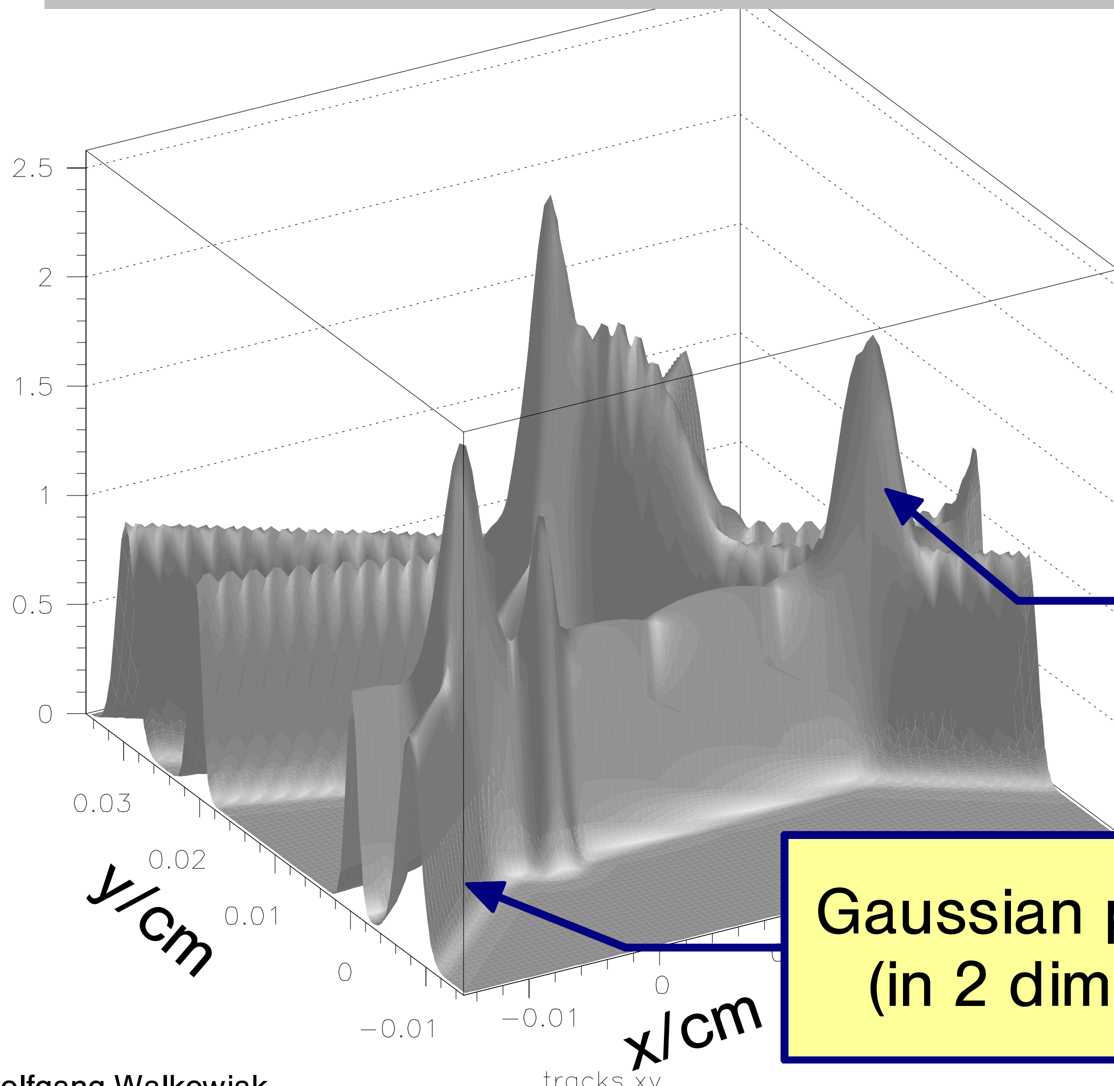
Functional principle

- ZVTOP operates on tracks from a jet:
  - Probability tubes formed from tracks and their covariance matrices for each track.
  - Overlaps of probability tubes determine vertex significance.
  - Initially two-track candidate vertices are formed.
  - Candidate vertices are checked for being resolved from each other and merged if not.
  - Tracks are assigned to vertices according to vertex significance.
  - Final list of vertices is formed and position is fitted by employing a track fitter to tracks in vertex.

Probability tubes

Example:

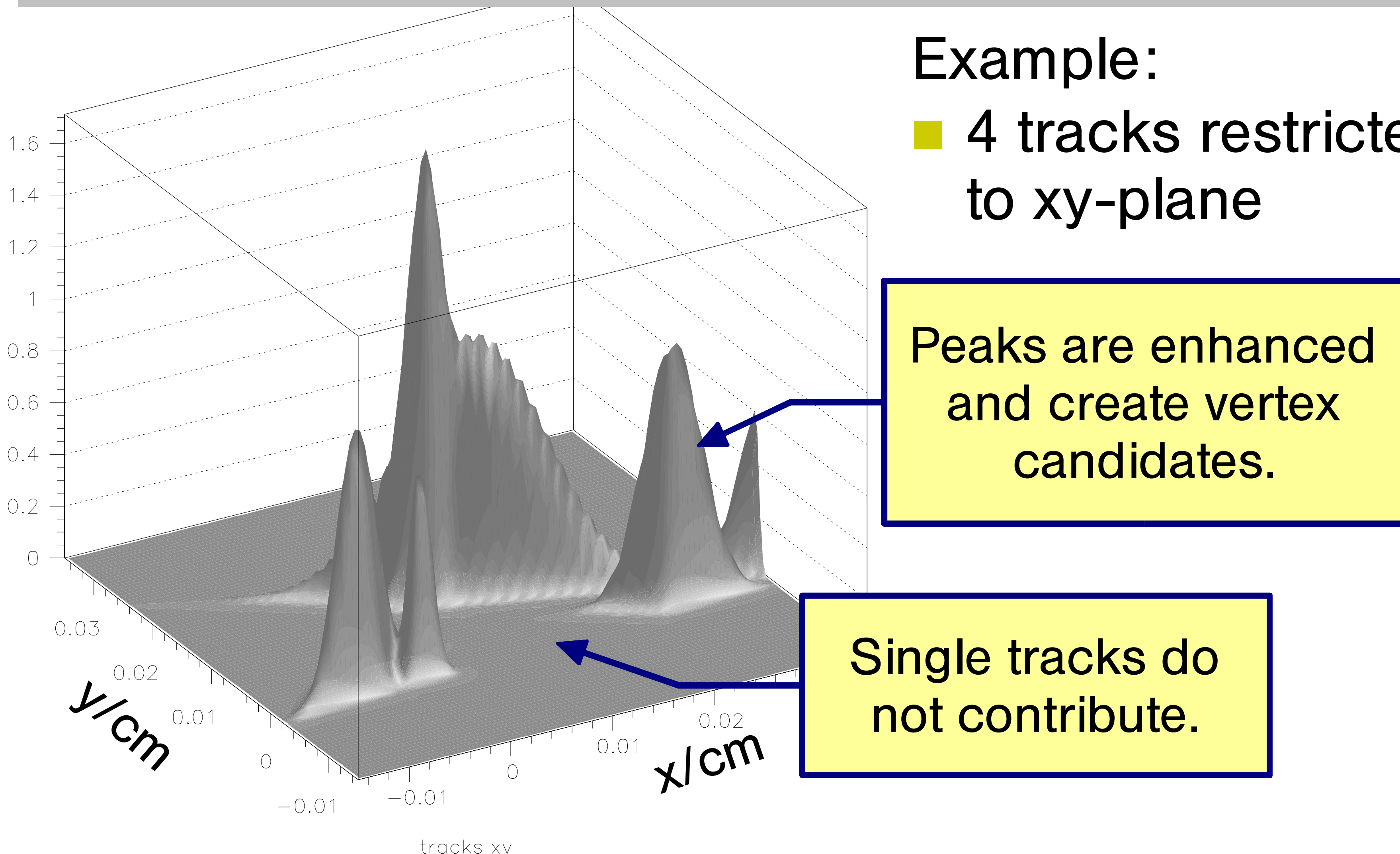
- 4 tracks restricted to xy-plane



Peaks at crossings of probability tubes

Gaussian probability (in 2 dimensions)

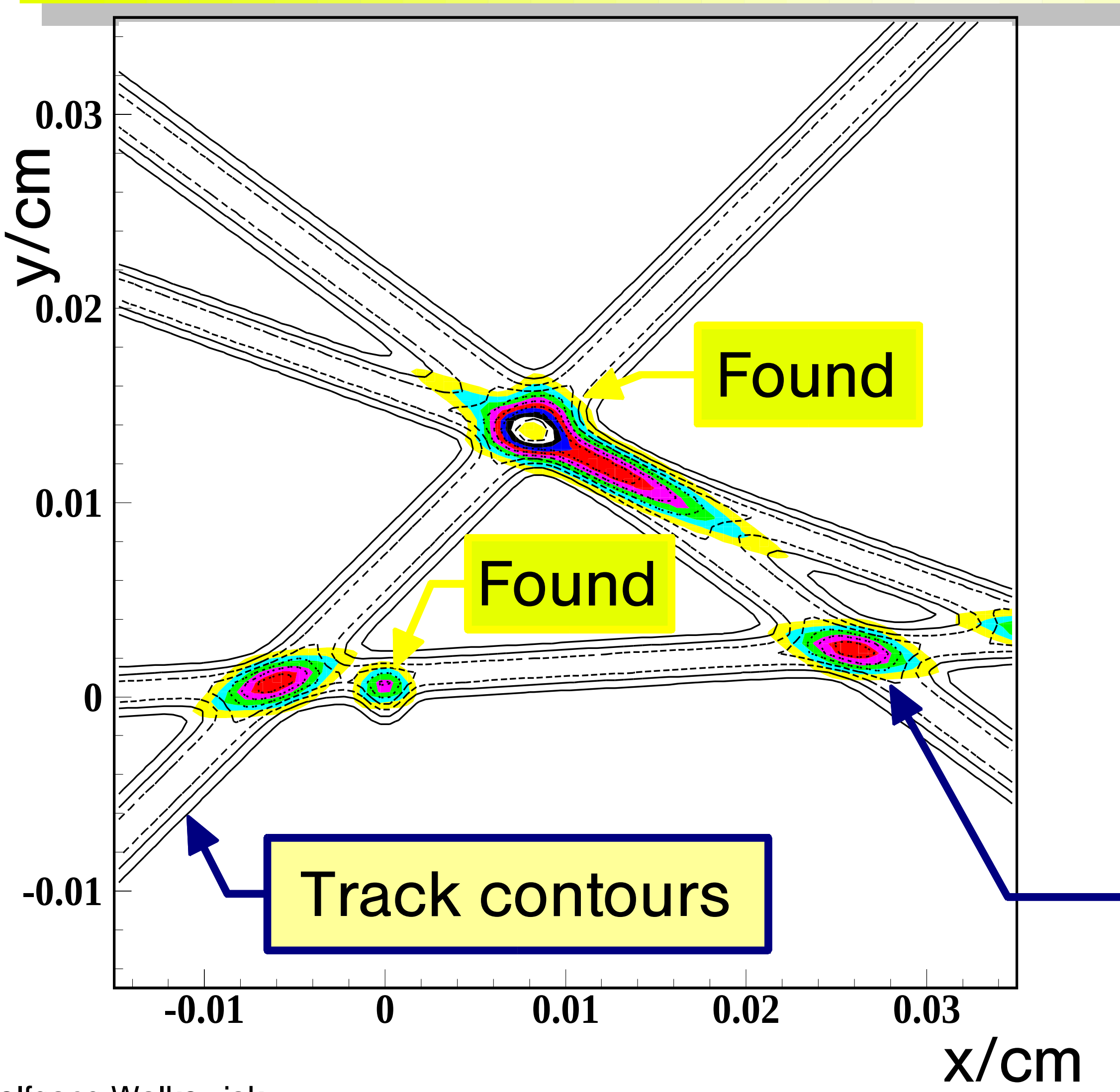
Vertex significance



Example:

- 4 tracks restricted to xy-plane

Vertex finding result



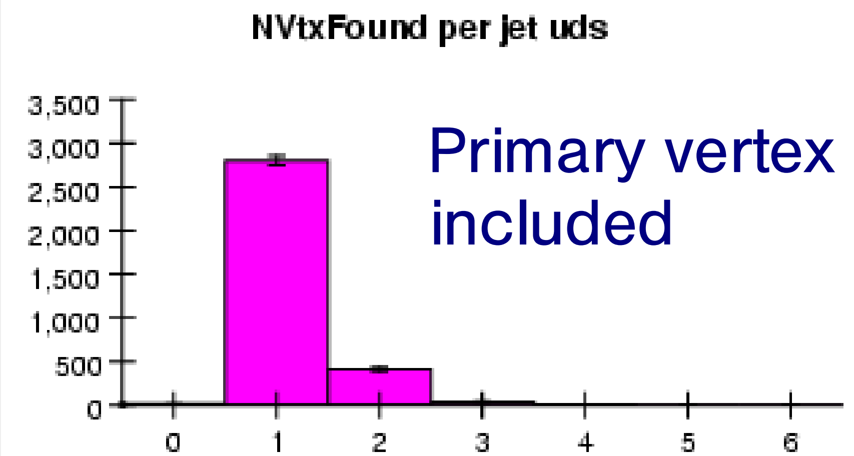
Example:

- 4 tracks restricted to xy-plane

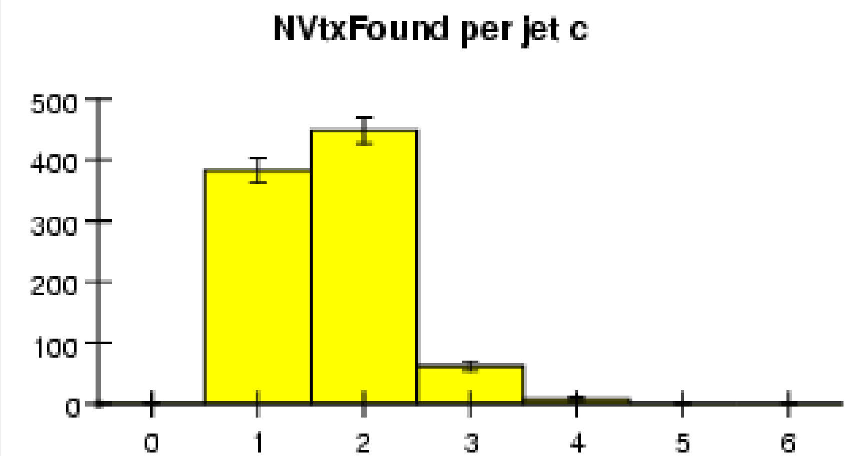
Found vertices:

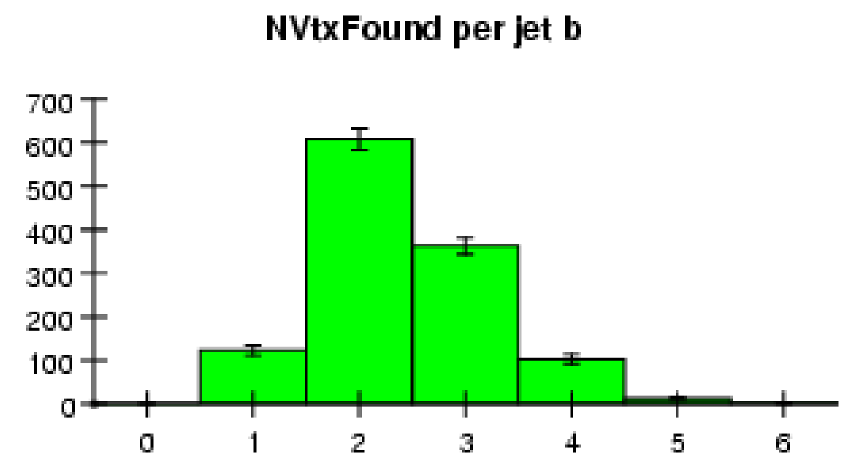
- Vertex with 3 tracks
- Vertex with IP + 1 tracks

Vertex finding in $q\bar{q}$ events



Sample:
2500 $q\bar{q}$ events at
500 GeV (Pythia)
($q = u,d,s,c,b$)





Status of Java implementation

ZvTopVertexer is coded and operational

- Finds vertices (verified with single events)
- Gives reasonable output for vertex multiplicities
- Checks of fitting results (deviation from MC origin, vertex fitting χ^2) need to be done systematically.
- Ghosttrack algorithm is not yet implemented.

Summary

Fast simulation framework in Java:

- Track smearing with full covariance matrices for S2 and L2 detector design available.
- Track smearing for other detector topologies is easily generated, allowing direct comparisons.
- Fast calorimeter simulation algorithm (as already implemented in ROOT) will be implemented.

Toplogical vertexing à la SLD's ZVTOP will (soon) be available in Java/JAS framework.

Conclusions

- The Java Analysis Studio provides environment for fast and flexible simulation studies.
- Different detector scenarios can be quickly modelled and sophisticated physics analyses can be carried out (e.g. topological vertexing).
- We plan to extend the functionality and sophistication for simulations (e.g. calorimeter).
- We encourage your input, both comments and contributed effort.