Vertex 2008

17th International Workshop on Vertex Detectors

July 28–August 1, 2008, Utö Island, Sweden

The LCFIVertex package

- Overview of the LCFIVertex package
- Recent extensions
- ZVMST: a minimum spanning tree-based vertex finder
- Summary

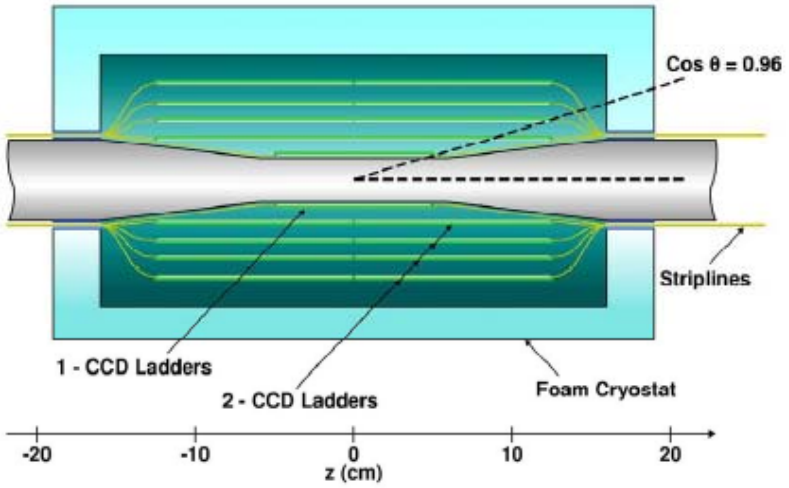
Sonja Hillert (University of Oxford)

Introduction

- **The LCFIVertex package provides:**
  - **vertex finder ZVTOP: branches ZVRES, ZVKIN (new in ILC environment), ZVMST (new)**
  - **flavour tagging: neural net approach (algorithm: R. Hawkings, LC-PHSM-2000-021); includes neural net package; permitting change of inputs, network architecture**
  - **quark charge determination**

- **software was developed in the context of vertex detector R&D for the International Linear Collider, building on earlier work at SLD;**
- **used for detector optimisation for the ILC-detector LoIs due in spring 2009;**
- **generically applicable for low-mass vertex detectors of high point resolution**

- **example ILC vertex detector design:**
  - **angular coverage to cos θ = 0.96**
  - **5 layers from 15mm to 60 mm radius**
  - **layer thickness 0.1% X_0**
  - **point resolution ~ 3 μm**



LCFIVertex and new vertex finder ZVMST

- **first version of LCFIVertex released April 2007:**

  code, default flavour tag networks and documentation available from ILC software portal http://www-flc.desy.de/ilcsoft/ilcsoftware/LCFIVertex

- **since then numerous extensions;**

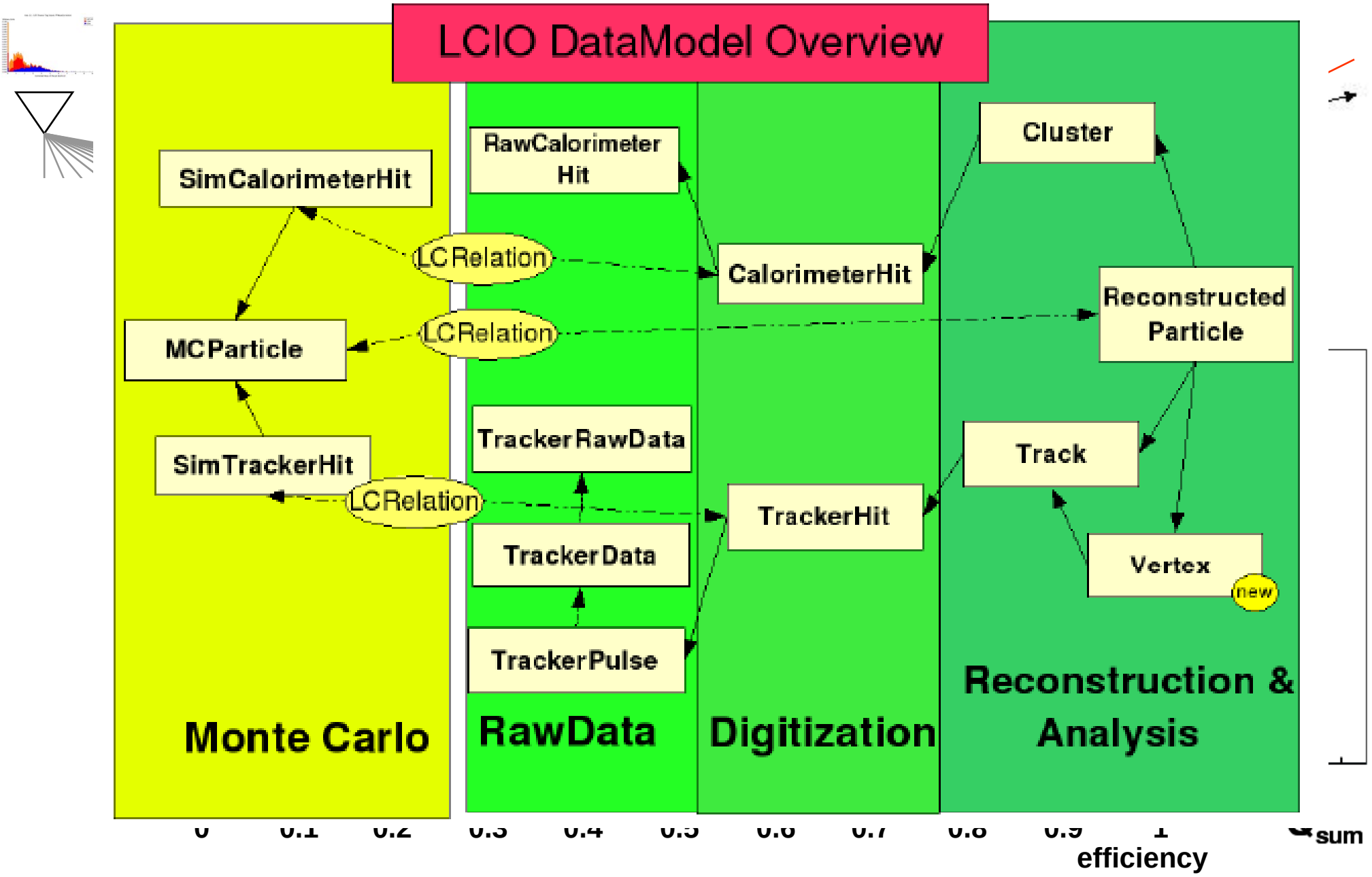
  in this talk focus on new vertex finder ZVMST (soon to be added to LCFIVertex-CVS);

- **ZVMST based on minimum spanning trees: mathematical optimisation tool with a wide range of applications, e.g. source detection in gamma ray images**

- **Method exploits topological information (e.g. connectedness of detected photons in astrophysical example) → provides natural approach to topological vertex finding**

- **NIM paper on LCFIVertex package in preparation**

  LC-note on ZVMST (approach and performance study) submitted


LCIO DataModel Overview
SimCalorimeterHit
RawCalorimeter Hit
Cluster
LCRelation
CalorimeterHit
Reconstructed Particle
MCParticle
LCRelation
SimTrackerHit
TrackerRawData
Track
LCRelation
TrackerHit
TrackerData
Vertex
new
TrackerPulse
Monte Carlo
RawData
Digitization
Reconstruction & Analysis
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
sum
efficiency

Extensions since the first release

- **ConversionTagger: identifies tracks from K_S, Λ decays , photon conversions *(K. Harder)***
- **Flavour tag: fit macro finds parameters for joint probability *(E. Devetak)***
- **Vertex charge: now separate processor, extensions in preparation *(E. Devetak)***
- **Detailed diagnostic tools (LCFIAIDAPlotProcessor): *(V. Martin)***
  - **Plots of flavour tag inputs and NN-outputs (global & for 1-, 2-, 3-vertex case);**
  - **Tables of efficiency & purity as function of neural network output;**
  - **Graphs of purity vs efficiency and mistag rates vs efficiency for all tags**
- **Kalman filter based vertex fitter (Gorbunov, Kisel; interfaced by *T. Lastovicka*)**
- **Compatibility with DST-format as agreed by ILD detector concept *(C. Lynch)***
- **Tuning of track selection and code parameters, nearly complete *(R. Walsh)***
- **Training of new set of neural networks with optimised configuration *(R. Walsh)***
- **ZVMST: new vertex finder based on minimum spanning tree *(S. Hillert)***
- **"Vertex Cheater" using track-combinations from MC, realistic fitter *(S. Hillert)***

The ZVMST algorithm

- uses same mathematical description of topological information that was carefully developed by D. Jackson for ZVTOP_ZVRES (NIM A 388 (1997) 247)

- differs in the way this information is extracted and used to find vertices

  - ZVRES: iterative approach that performs finding the correct vertex *positions* in conjunction with finding the correct *track combinations*
  - ZVMST: finds near-final positions, then assigns tracks to them, finally performs fit

- Next slides:

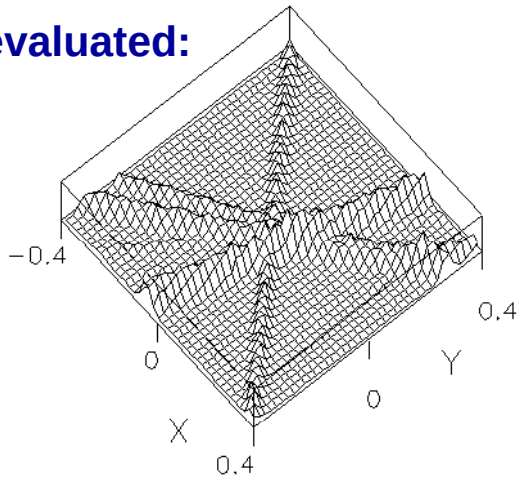
  - Mathematical representation of topological input: track functions and vertex function
  - Minimum spanning trees
  - Description of ZVMST algorithm
  - Vertex finding for an example input jet:
    - ZVRES
    - ZVMST

Track probability tubes and Vertex Function

- **Central idea of ZVTOP:** describe tracks by probability density functions and combine them to form a vertex function encoding the topological information for a jet

- **Track probability functions:** Gaussian profile in the plane normal to trajectory at point of closest approach $\vec{p}$ to 3D-space point $\vec{r}$ at which function is evaluated:

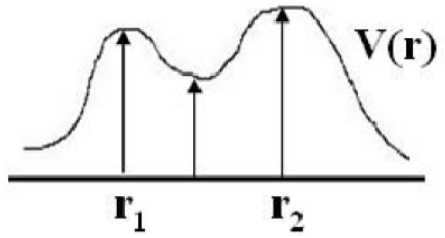
$$f_i(\vec{r}) = \exp\left\{ -\frac{1}{2} (\vec{r} - \vec{p}) \, \mathbb{V}_i^{-1} (\vec{r} - \vec{p})^T \right\}$$



- **Vertex function:** simplest form: $V(\vec{r}) = \sum_{i=1}^{N} f_i(\vec{r}) - \frac{\sum_{i=1}^{N} f_i^2(\vec{r})}{\sum_{i=1}^{N} f_i(\vec{r})}$

  can be extended by a contribution for the IP, and weighted such that it is smaller at large angles to the jet axis (to suppress fake vertices)

- **Two main purposes of vertex function in ZVTOP_ZVRES** (classical vertex finder):
  - Identify space points at which vertices are likely to be found
  - Decide on whether two vertex candidates are resolved



Minimum spanning trees

- Mathematically, a minimum spanning tree is a special type of graph.
- A graph is a set of nodes that are connected by edges.
- Each edge can have a weight assigned to it.

- Trees are graphs in which the edges do not form loops. Thus, For a graph with loops there exist several trees.
- The minimum spanning tree is the tree that minimises the sum of the weights of the edges.
- If none of the weights are identical, there always exists exactly one minimum spanning tree.

- Efficient algorithms for finding the minimum spanning tree of a given graph exist, e.g. Dijkstra's algorithm, Kruskal's algorithm.
- Well tested, optimised implementations of these algorithms are available in the boost library.

ZVMST algorithm

- Find all two-track combinations with V(r) > 0.001, $\chi^2 < 10$
- Set up a graph that contains all these combinations as edges, with 1/V(r) as weights
- Find the minimum spanning tree for this graph [minimises the weights, maximises V(r)]
- Minimum spanning tree returns list of edges, sorted such that largest V(r) first in list
- For each of the corresponding candidate vertices, find r_{MAX} of max' V(r) in vicinity;
- Keep only candidates with r_{MAX} values > 400 μm from already selected ones
- Assign tracks to the selected positions based on Gauss-tube values and $V(r_{MAX})$
- Try to reassign tracks for which no other track was assigned to same candidate position
- Sort wrt distance from IP, add separate IP-vertex, if closest is > 600 μm apart from IP

ZVMST: criteria for assigning tracks to vertices

Further details on the step described as "Assign tracks to the selected positions based on Gauss-tube values and $V(r_{MAX})$":

- **For each track and each selected 3D position calculate Gauss tube value $f_i(r_{MAX})$ and $V(r_{MAX})$**
- **Find the two positions with the largest Gauss tube: $f_i(r_{i1}) > f_i(r_{i2})$ {by definition}**

- **If $f_i(r_{i2}) < f_{i,min}$: track has sizeable tube value only near one candidate → assign if $f_i(r_{i1}) > f_{i,min}$**
- **If $V(r_{i1}) > V(r_{i2})$: both tube value and vertex function maximal near same candidate**

  →assign to r_{i1} if $f_i(r_{i1}) > f_{i,min}$
- **If $V(r_{i1}) < V(r_{i2})$, calculate relative differences**

$$\Delta f_{i1,2} = \frac{|f_i(\vec{r}_{i1}) - f_i(\vec{r}_{i2})|}{f_i(\vec{r}_{i1}) + f_i(\vec{r}_{i2})} \quad \text{and} \quad \Delta V_{i1,2} = \frac{|V_i(\vec{r}_{i1}) - V_i(\vec{r}_{i2})|}{V_i(\vec{r}_{i1}) + V_i(\vec{r}_{i2})}$$

If relative difference in tube values is "not too large" and relative difference in vertex function is "large enough", assign to r_{i2} (more precisely, if $\Delta f_{i1,2} < \Delta f_{max}$ and $\Delta V_{i1,2} > \Delta V_{min}$)

Example – MC truth

Particles corresponding to tracks have following MC origins:

Track	x	y	z
0	0	0	0
1	-0.251	-2.146	2.046
2	0	0	0
3	-0.405	0.030	0.177
4	0	0	0
5	0	0	0
6	-0.405	0.030	0.177
7	-0.405	0.030	0.177
8	-0.405	0.030	0.177
9	0	0	0
10	-22.920	-210.4	200.7
11	-0.251	-2.146	2.046
12	0	0	0

COLOUR CODE: **blue: IP-tracks, green: true 2nd vertex, red: true 3rd vertex**

Example – ZVRES

- Find all two-track combinations and 1-track-IP combinations with V(r) > 0.001, $\chi^2 < 10$
- For each track, find the candidates it is contained in and sort them with respect to V(r)

0	1	2	3	4	5	6	7	8	9	10	11	12
(0, IP)	(1, 11)	(2, IP)	(3, 8)	(4, IP)	(5, IP)	(3, 6)	(7, 12)	(8, IP)	(9, IP)		(9, 11)	(12, IP)
											(1, 11)	
						(4, 6)						

- For each track remove candidates with V(r) < 10% of maximum V(r) for that track's candidates
- For each track retain track only in candidates that are resolved from each other

Example – ZVRES

- Sort all remaining candidates with respect to maximum vertex function value in the vicinity of the fitted vertex position $V(r_{MAX})$:
- Form clusters of unresolved candidates, and merge them
- Trim by χ^2 (i.e. remove tracks with χ^2 contribution > 10)
- Consider candidates in order of $V(r_{MAX})$ & remove tracks from candidates with lower $V(r_{MAX})$
- Add IP, if not yet contained and sort with respect to distance from IP

(1, 11)
(6)
(3)
(6)
(7)
(11)
(0)
(2)
(4)
(5)
(8)
(9)
(12)

(0, 2, 4, 5, 7, 8, 9, 12)

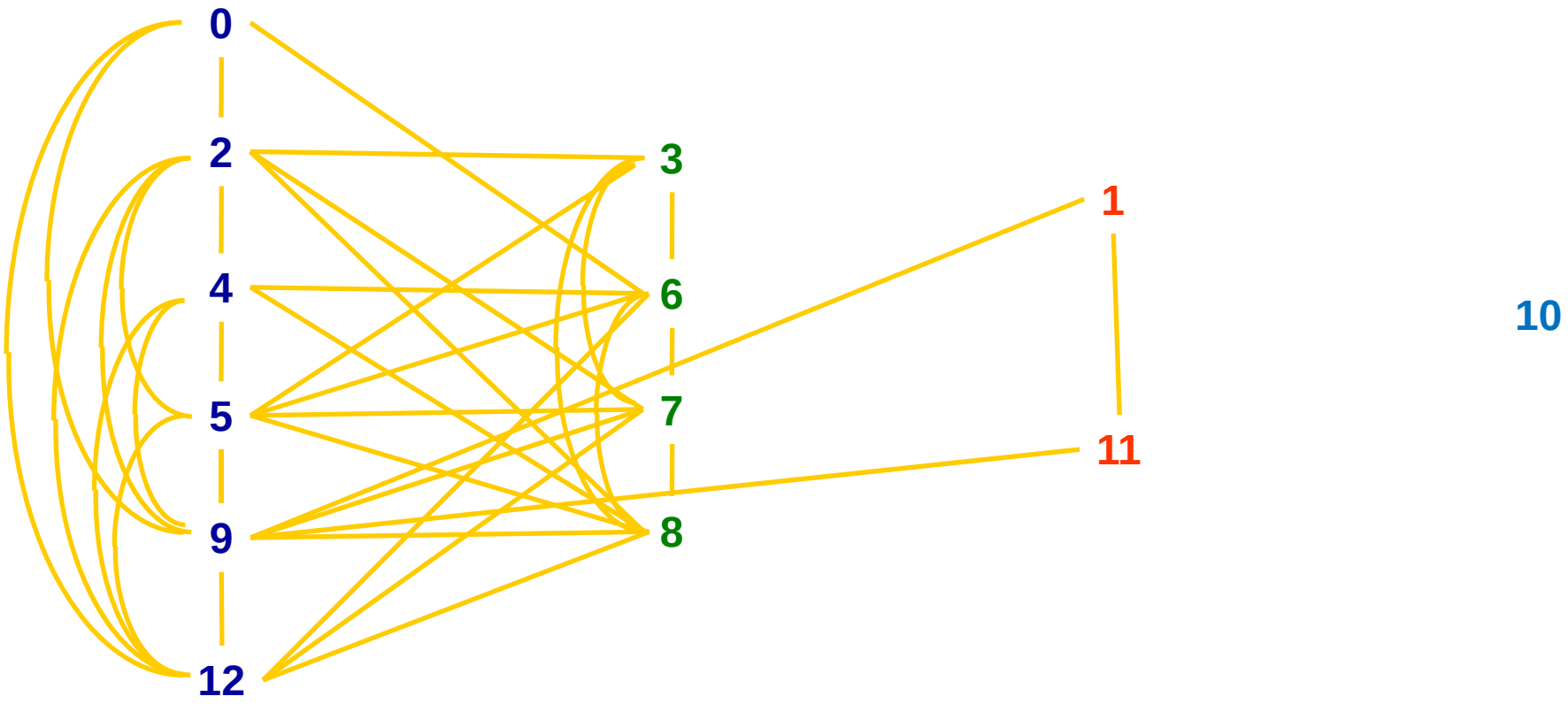
(1, 11)

(0, 2, 4, 5, 7, 8, 9, 12)

(1, 11)

Example – ZVMST

- Find all two-track combinations with V(r) > 0.001, $\chi^2 < 10$
- Set up a graph that contains all these combinations as edges, with 1/V(r) as weights
- Find the minimum spanning tree for this graph [minimises the weights, maximises V(r)]



Example – ZVMST

- Minimum spanning tree returns list of edges, sorted such that largest V(r) first in list
- For each of the corresponding candidate vertices, find r_{MAX} of max' V(r) in vicinity;
- Looping over cand's, keep only those with r_{MAX} values > 400 μm from already selected ones
- Assign tracks to the selected positions based on Gauss-tube values and $V(r_{MAX})$
- Try to reassign tracks for which no other track was assigned to same candidate position
- Sort wrt distance from IP, add separate IP-vertex, if closest is > 600 μm apart from IP

(4, 8)	(4, 8) : (-0.010, -0.009, 0.012)	(0, 2, 4, 5, 7, 8, 9, 12)	(0, 2, 4, 5, 7, 8, 9, 12)
(4, 12)			(3, 6)
(2, 12)			
(5, 12)			
(9, 12)			
(0, 12)			
(7, 12)			
(3, 8)	(3, 8) : (-0.405, 0.030, 0.176)	(3, 6)	
(3, 6)			
(9, 11)	(9, 11) : (-0.371, -2.450, 2.286)		
(1, 11)	(1, 11) : (-0.207, -1.931, 1.856)		

Performance study

- following slides: performance comparison between ZVRES and ZVMST in terms of
  - Vertex multiplicity (inclusive and as function of track multiplicity)
  - "purity" of reconstructed vertices – have the correct tracks been combined?
  - resulting flavour tag performance

- Vertex cheater written to provide a reference for the reconstruction algorithms

- cheater uses perfect track-to-vertex assignment based on MC information

  (NOTE: to be understood within the limitations of jet-based approach,

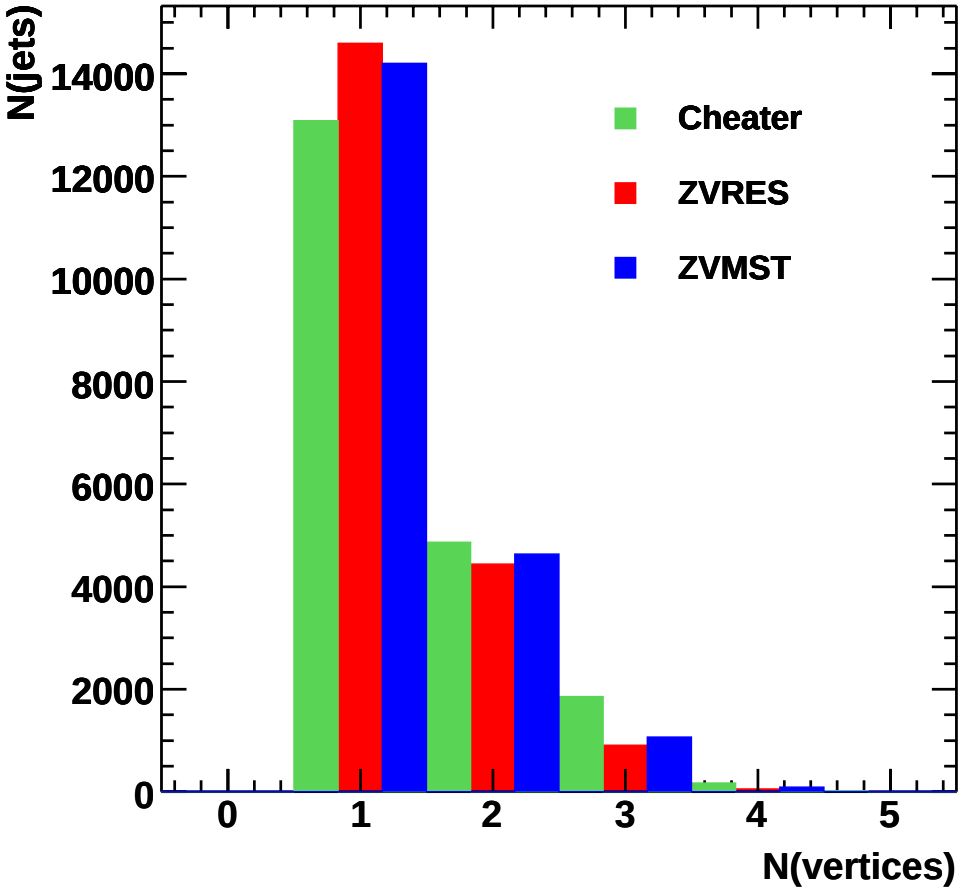
  i.e. if tracks from 2-prong vertex end up in different jets, vertex not considered)

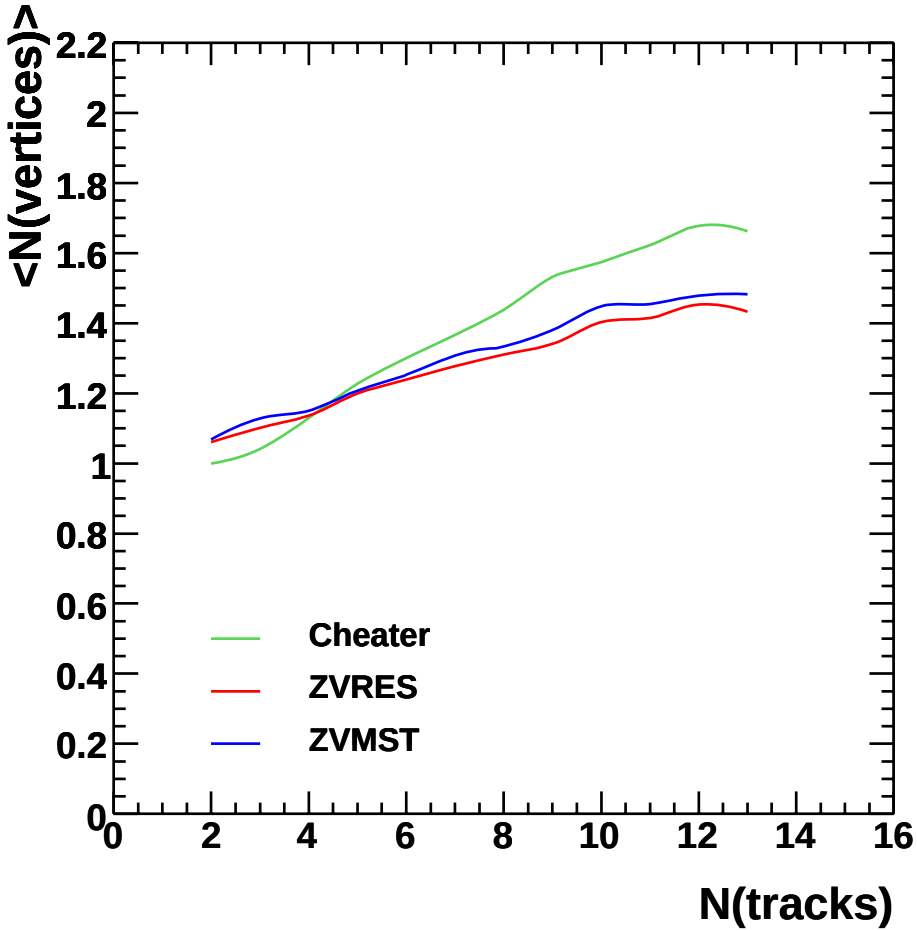
- Sets of tracks obtained from MC-information passed to vertex fitter in the usual way

  →disentangle problem of track-to-vertex assignment and finding correct position

- Resulting vertices can be treated the same way as reconstructed ones in the subsequent steps (e.g. flavour tag inputs calculation)

Vertex Multiplicity





- Both algorithms find smaller number of vertices than present in MC, ZVMST finds slightly more
- In particular, as the track multiplicity increases the average number of found vertices does falls short of the true MC number by an increasing amount

Purity of track-to-vertex assignment, b-jets

Monte Carlo track origin		Reconstructed track-vertex association Two vertex case (b)			Three vertex case (b)			
		pri	sec	iso	pri	sec	ter	iso
Primary	Cheater	93.8	0.508	24.2	98.7	0.167	1.29	46.8
	ZVRES	89.8	1.83	34.3	94.4	5.85	5.74	47.5
	ZVMST	89.1	2.31	46.6	95.5	5.78	5.85	57.7
B decay	Cheater	5.95	41.2	41.6	1.24	79.8	14.5	28.8
	ZVRES	7.26	48.3	30.5	3.83	65.5	12.9	25.6
	ZVMST	8.05	48.5	22.6	3	67.3	17.4	19.7
D decay	Cheater	0.242	58.3	34.2	0.0295	20	84.2	24.4
	ZVRES	2.92	49.8	35.2	1.74	28.7	81.3	26.9
	ZVMST	2.81	49.2	30.8	1.53	27	76.7	22.5
all above	Cheater	52.4	31.1	16.5	40.7	28.8	22.7	7.8
	ZVRES	49.3	36.9	13.8	38.2	29.7	23.4	8.73
	ZVMST	45.5	35.3	19.2	34.2	28.3	23.7	13.7

ZVMST closer to Cheater than ZVRES

ZVRES closer to Cheater than ZVMST

- **Improvement for IP-vertex and secondary in 3-vertex b-jets**
- **degradation for 2-vertex-b-jets**

Purity of track-to-vertex assignment, c-jets

Monte Carlo track origin		Reconstructed track-vertex association						
		Two vertex case (c)			Three vertex case (c)			
		pri	sec	iso	pri	sec	ter	iso
Primary	Cheater	99.8	0.862	74.4	99.9	7.16	26.7	75.9
	ZVRES	94.8	7.17	75	94.3	26.7	33.6	67.4
	ZVMST	95.8	9.2	78.2	95.7	21.2	44.5	73.8
D decay	Cheater	0.191	99.1	25.6	0.104	92.8	73.3	24.1
	ZVRES	5.23	92.8	25	5.67	73.3	66.4	32.6
	ZVMST	4.18	90.8	21.8	4.29	78.8	55.5	26.2
all above	Cheater	65.4	27.2	7.44	56.3	19.7	18.9	5.11
	ZVRES	64.1	27.3	8.59	49.5	22.3	21.1	7.1
	ZVMST	60.9	27.7	11.4	44.9	23	20.1	11.9

ZVMST closer to Cheater than ZVRES

ZVRES closer to Cheater than ZVMST

- overall improvement for c-jets
- degradation of purity for vertex furthest from IP
- note that increase in purity might partly be due to fewer tracks being assigned compared to ZVRES

Reasons for remaining confusion for vertex cheater

➢ **Track-purities for the vertex cheater are not always 100% or 0; reasons include:**

- **Non-IP one-prong vertices cannot be found by either of the reconstruction algorithms and have therefore intentionally been treated as non-reconstructable in the cheater; intentional, so the cheater can give an idea of best performance under this boundary condition**

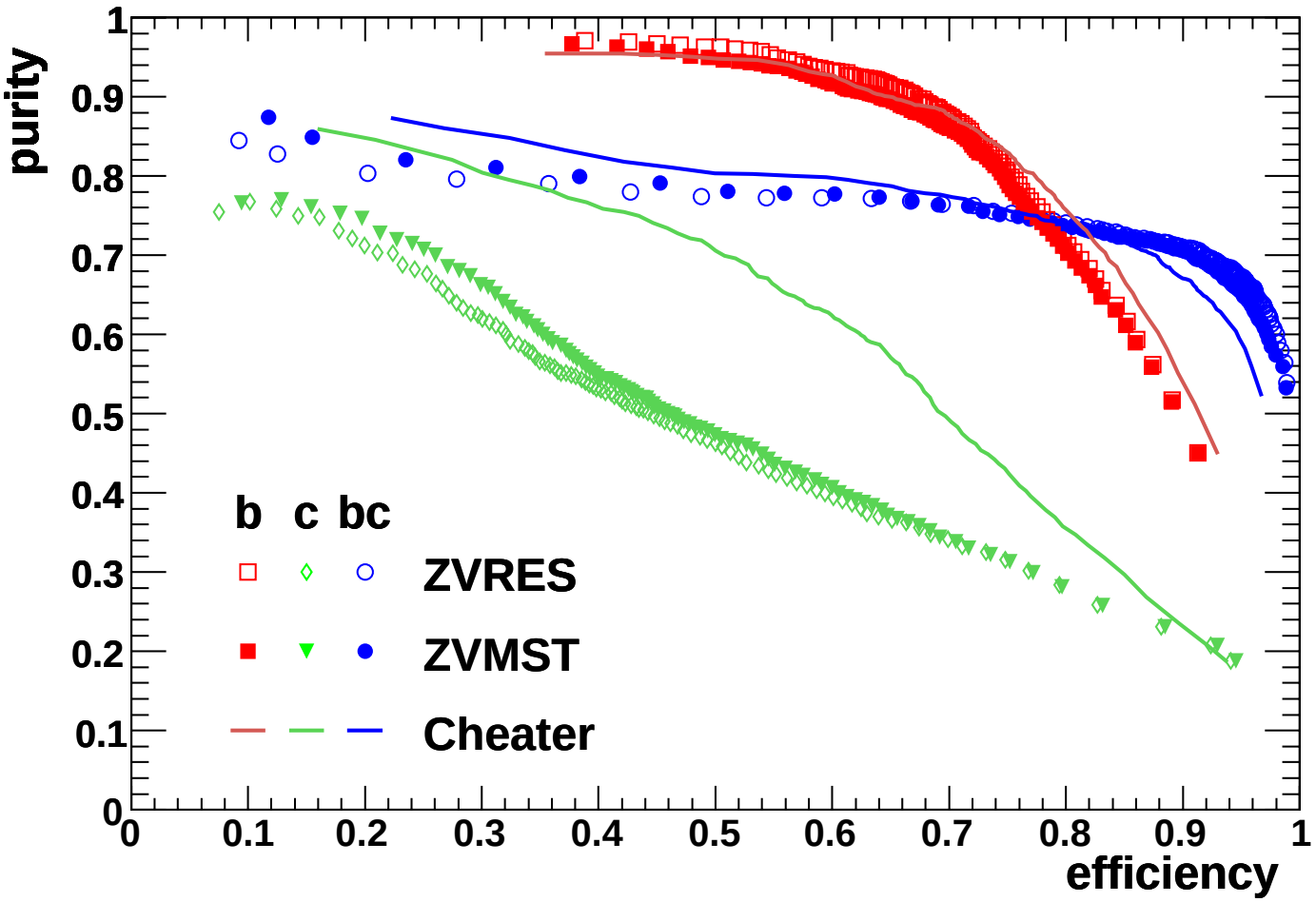
- **For reconstructed vertices, the decision which of the found vertices is the primary, secondary or tertiary is taken on the basis of their distance from the IP (also for cheater), while for the MC-origin the MC-parentage is decisive, even if D-decay closer to IP than B-decay**

- **Hadronic interactions add to confusion in correspondence between 1st, 2nd and 3rd vertex from IP and primary, B-decay and D-decay vertex**

- **Due to effects of pattern recognition, correspondence between tracks and MC particles is imperfect**

➢ **cheater provides a guideline what to expect for the reconstruction**

Resulting flavour tagging performance



- ZVMST yields improved c-tag purity over full efficiency range,
- up to 5% at low efficiency;
- slight degradation (up to 1.5%) for b-tag;

- Cheater comparison:
  - If further improvements in track-to-vertex assignment could be made, this should directly improve c-tag
  - b-tag close to optimal

NOTE: cheater only shows “optimum” achievable with current track selection, flavour tag approach and SGV-nets

Conclusions

- **The LCFIVertex package is an essential tool for preparation of detector LoIs for the ILC, with generic applicability to high-precision vertex detectors.**

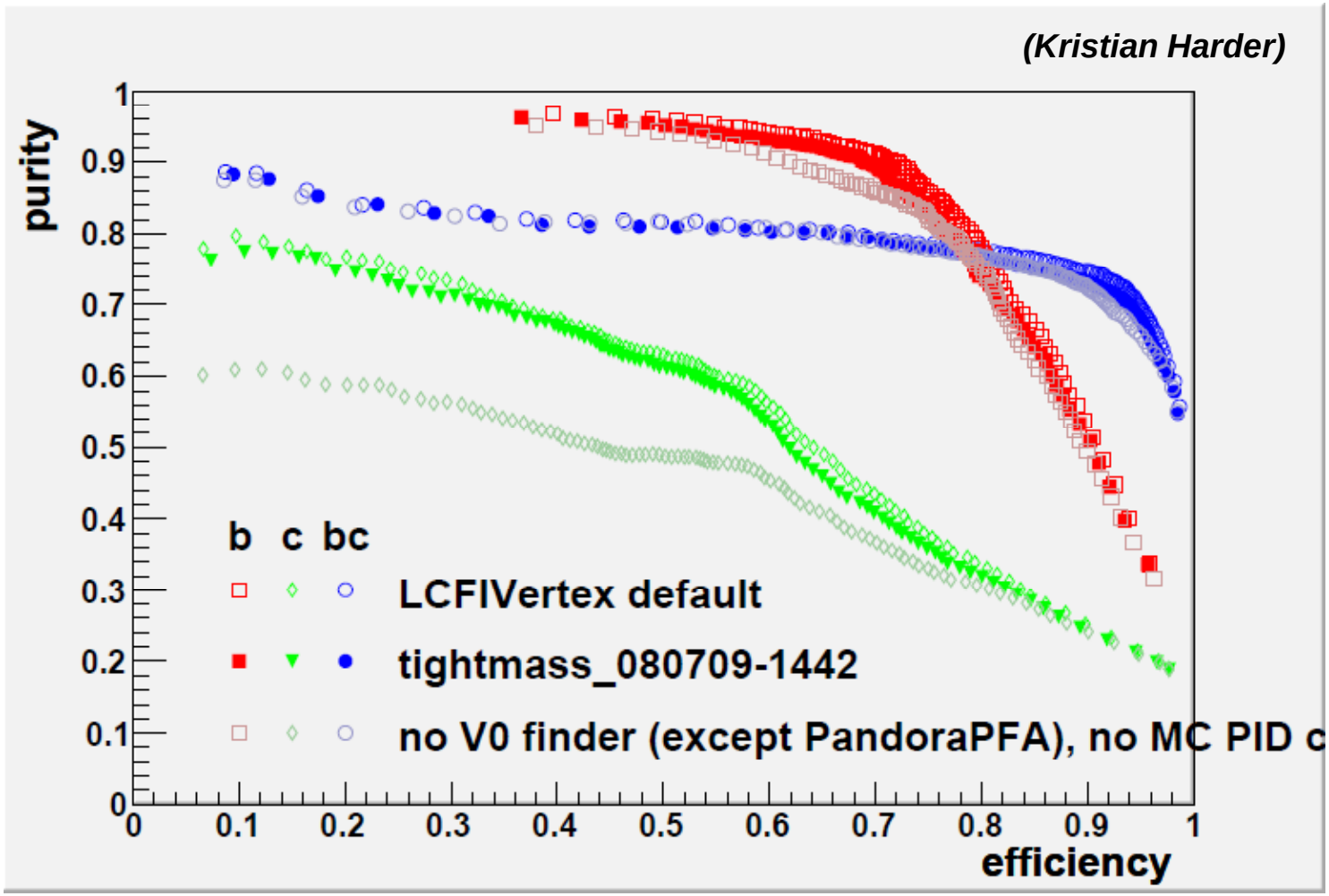
- **Additions since the first release in 2007 include extensive diagnostics, technical improvements, work towards default configuration, and the new vertex finder ZVMST.**

- **ZVMST results are competitive with those obtained from ZVRES:**
  - **Vertex multiplicities from ZVMST slightly closer to MC reference than for ZVRES.**
  - **Track-to-vertex assignment similar, some aspects improved, some degraded;**
  - **ZVMST yields up to 5% increased c-tag purity, up to 1.5% lower b-tag purity when using the same flavour tagging approach and networks from SGV**

- **NOTE: algorithm parameters not yet optimised for either approach (although ZVRES parameters optimised previously, and a preliminary study was performed for ZVMST)**

Additional Material

ConversionTagger – flavour tag result



Flavour tag improvement

- improvement to calculation of one of the inputs for the flavour tag-NN:

- probability that a track from primary vertex has impact parameter significance > b/σ_b is

$$P_i = \frac{\int_{b/\sigma_b}^{\infty} f(x)\,dx}{\int_0^{\infty} f(x)\,dx}$$

- the symmetric function $f(x)$ needs to be obtained from the negative side of the b/σ_b distribution;
- first release of the Vertex Package used hard coded parameterisation obtained from fast MC
- Erik Devetak has written a module to obtain parameters properly from a fit

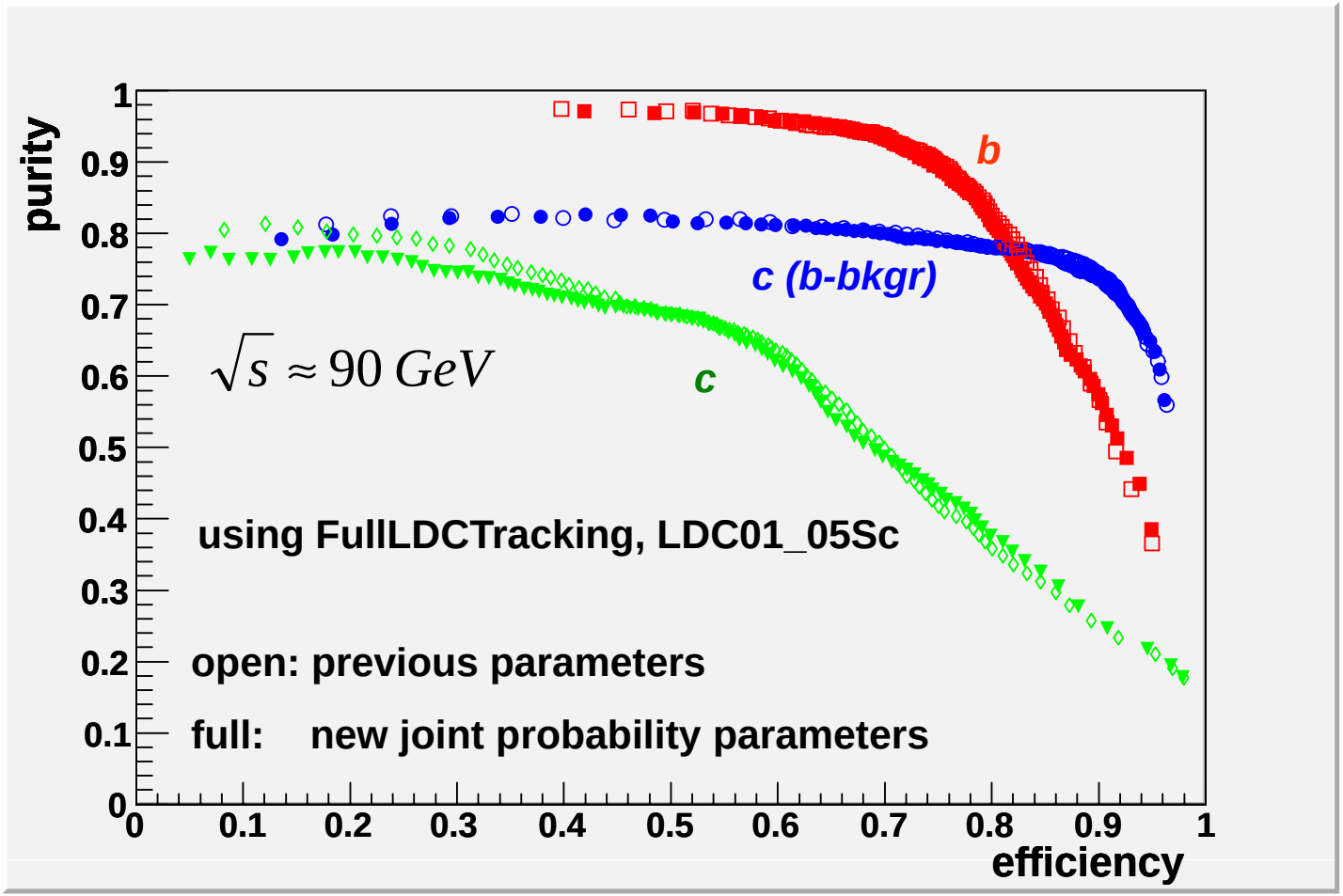
- joint probability (flavour tag input) for ensemble of tracks to come from primary vertex is

$$P_J = y \sum_{m=0}^{N-1} \frac{(-\ln y)^m}{m!}$$

Fit macro for joint probability calculation: results

parameters for Rϕ joint probability	
previous	new
1.013	1.015
0.025	0.280
0.102	0.561
0.041	0.006
0.016	0.048

parameters for Z joint probability	
previous	new
1.016	1.043
0.027	0.271
0.095	0.468
0.041	0.006
0.015	0.048



Obtained new values with full MC & reconstruction (left):

values change as expected, flavour tag performance stable.

ZVRES algorithm

- Find all two-track combinations and 1-track-IP combinations with V(r) > 0.001, $\chi^2 < 10$
- For each track, find the candidates it is contained in and sort them with respect to V(r)
- For each track remove candidates with V(r) < 10% of maximum V(r) for that track's candidates
- For each track retain track only in candidates that are resolved from each other
  (checking candidates in order of decreasing V(r), so the larger V(r) ones are retained)
- Sort all remaining candidates with respect to maximum vertex function value in the vicinity of the fitted vertex position $V(r_{MAX})$
- Form clusters of unresolved candidates, and merge them
- Trim by χ^2 (i.e. remove tracks with χ^2 contribution > 10)
- Consider candidates in order of $V(r_{MAX})$ & remove tracks from candidates with lower $V(r_{MAX})$
- Add IP, if not yet contained and sort with respect to distance from IP

Example 1 – MC truth

Particles corresponding to tracks have following MC origins:

Track	x	y	z
0	0	0	0
1	0	0	0
2	0.896	2.006	-0.156
3	0	0	0
4	0.319	0.823	-0.074
5	0.896	2.006	-0.156
6	0	0	0
7	0.319	0.823	-0.074
8	0.319	0.823	-0.074

COLOUR CODE: blue: IP-tracks, green: true 2nd vertex, red: true 3rd vertex

Example 1 – ZVRES

- Find all two-track combinations and 1-track-IP combinations with V(r) > 0.001, $\chi^2 < 10$
- For each track, find the candidates it is contained in and sort them with respect to V(r)

0	1	2	3	4	5	6	7	8
(0, IP)	(1, IP)	(2, 5)	(3, IP)	(4, 5)	(2, 5)	(6, IP)	(7, 8)	(7, 8)
				(4,)	(4, 5)			
				(4,)	(5, 7)		(5, 7)	

Note that for most tracks the highest V(r) candidate has tracks from same MC-vertex

- For each track remove candidates with V(r) < 10% of maximum V(r) for that track's candidates
- For each track retain track only in candidates that are resolved from each other

(checking candidates in order of decreasing V(r), so the larger V(r) ones are retained)

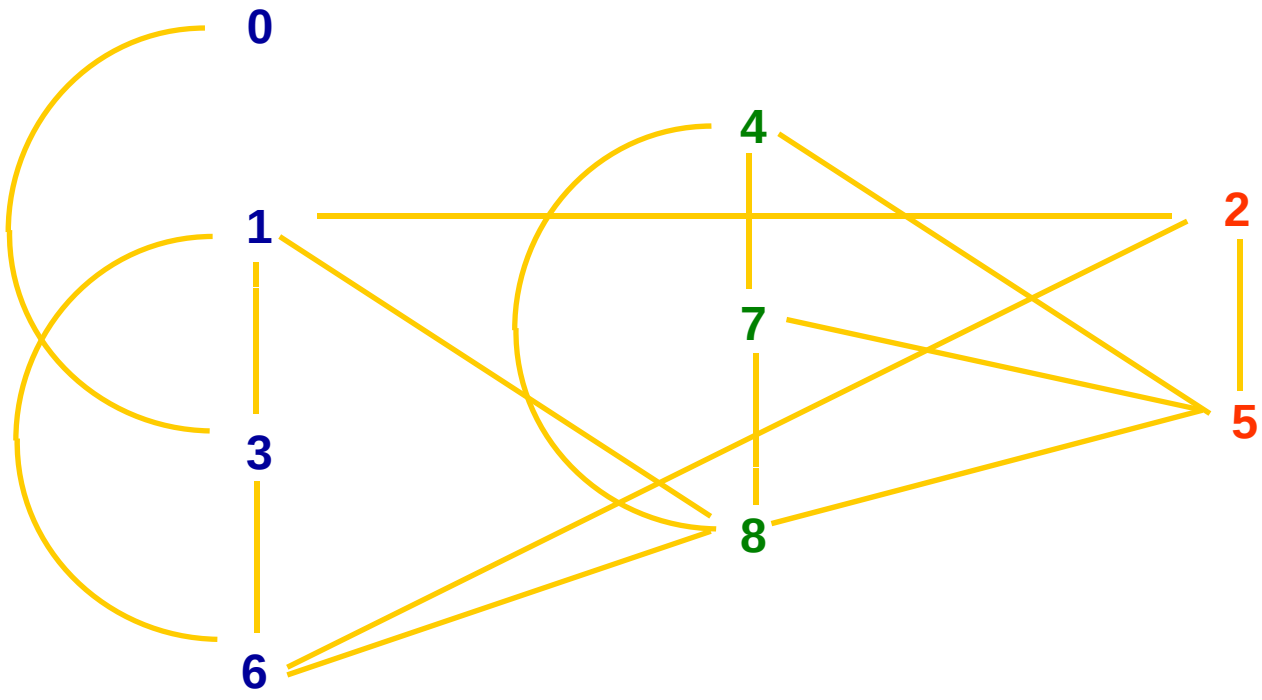
Example 1 – ZVRES

- Sort all remaining candidates with respect to maximum vertex function value in the vicinity of the fitted vertex position $V(r_{MAX})$:
- Form clusters of unresolved candidates, and merge them
- Trim by χ^2 (i.e. remove tracks with χ^2 contribution > 10)
- Consider candidates in order of $V(r_{MAX})$ & remove tracks from candidates with lower $V(r_{MAX})$
- Add IP, if not yet contained and sort with respect to distance from IP

(2, 5)	(0, 1, 3, 6)	(0, 1, 3, 6)
(4, 5)	(2, 5)	(4, 7, 8)
(4)	(4, 7, 8)	(2, 5)
(4)		
(5, 7)		
(7, 8)		
(0)		
(1)		
(3)		
(6)		

Example 1 – ZVMST

- Find all two-track combinations with V(r) > 0.001, $\chi^2 < 10$
- Set up a graph that contains all these combinations as edges, with 1/V(r) as weights
- Find the minimum spanning tree for this graph [minimises the weights, maximises V(r)]



Example 1 – ZVMST

- Minimum spanning tree returns list of edges, sorted such that largest V(r) first in list
- For each of the corresponding candidate vertices, find r_{MAX} of max' V(r) in vicinity;
- Looping over cand's, keep only those with r_{MAX} values > 400 μm from already selected ones
- Assign tracks to the selected positions based on Gauss-tube values and $V(r_{MAX})$
- Try to reassign tracks for which no other track was assigned to same candidate position
- Sort wrt distance from IP, add separate IP-vertex, if closest is > 600 μm apart from IP

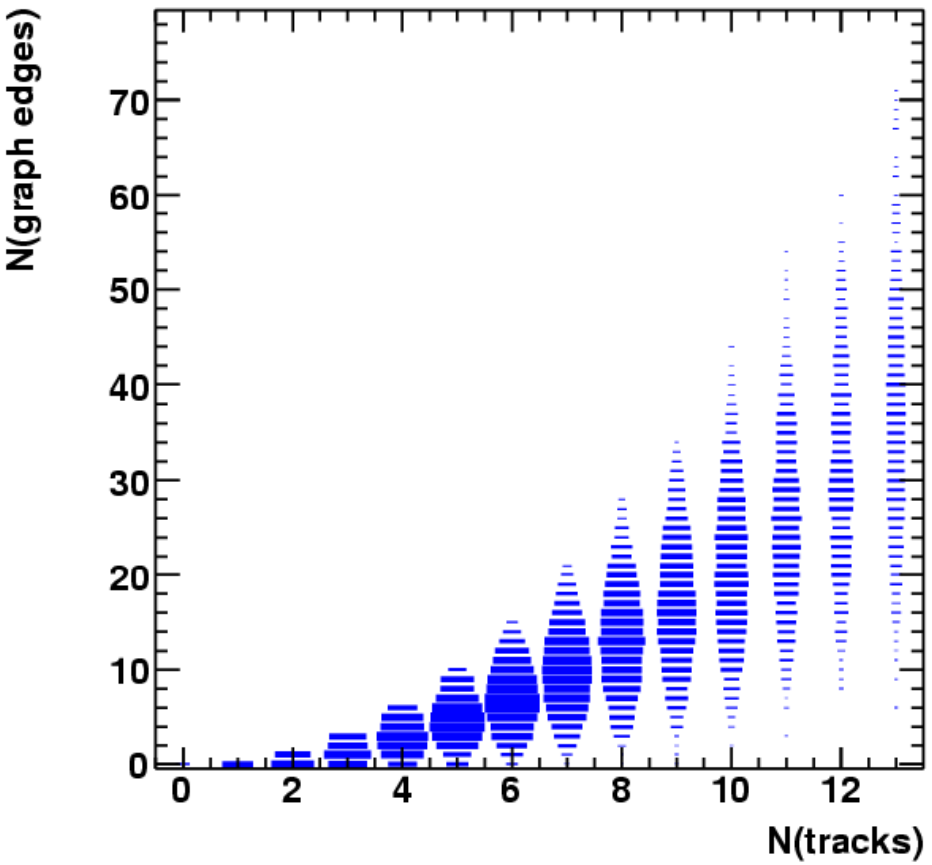
(0, 3)	(0, 3) : (-0.003, -0.000, 0.004)	(0, 1, 3, 6)	(0, 1, 3, 6)
(3, 6)			(4, 7, 8)
(1, 3)			(2, 5)
(2, 5)	(2, 5) : (0.832, 1, 889, -0.147)	(2, 5)	
(7, 8)	(7, 8) : (0.291, 0.743, -0.061)	(4, 7, 8)	
(4, 5)			
(4, 8)			
(1, 2)	(1, 2) : (0.203, 0.374, 0.117)		

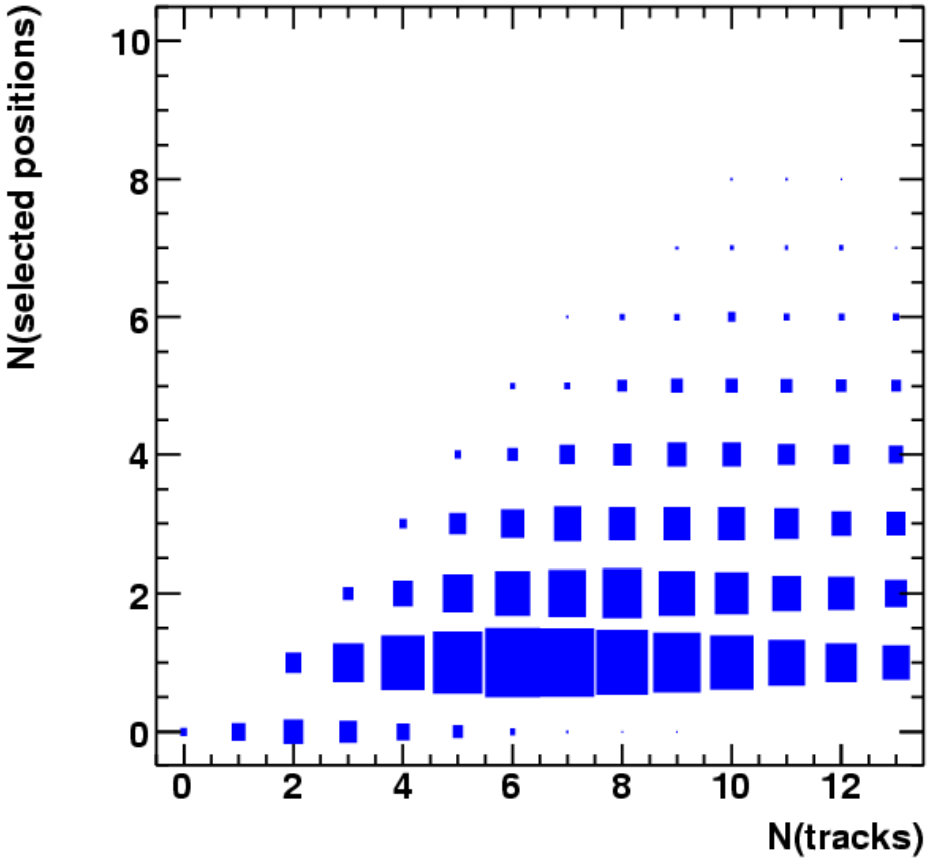
ZVMST – preliminary parameter tuning

- **Each of the ZVMST parameters varied while keeping the others fixed.**
- **Initial values picked "by hand" on the basis of detailed printout for 20 events.**
- **Resulting default values chosen on the basis of flavour tag purities at 70%, 80% and 90% efficiency.**
- **Note: sometimes b-, c- and bc-tag prefer different values, and sometimes at different efficiencies different values are preferred.**

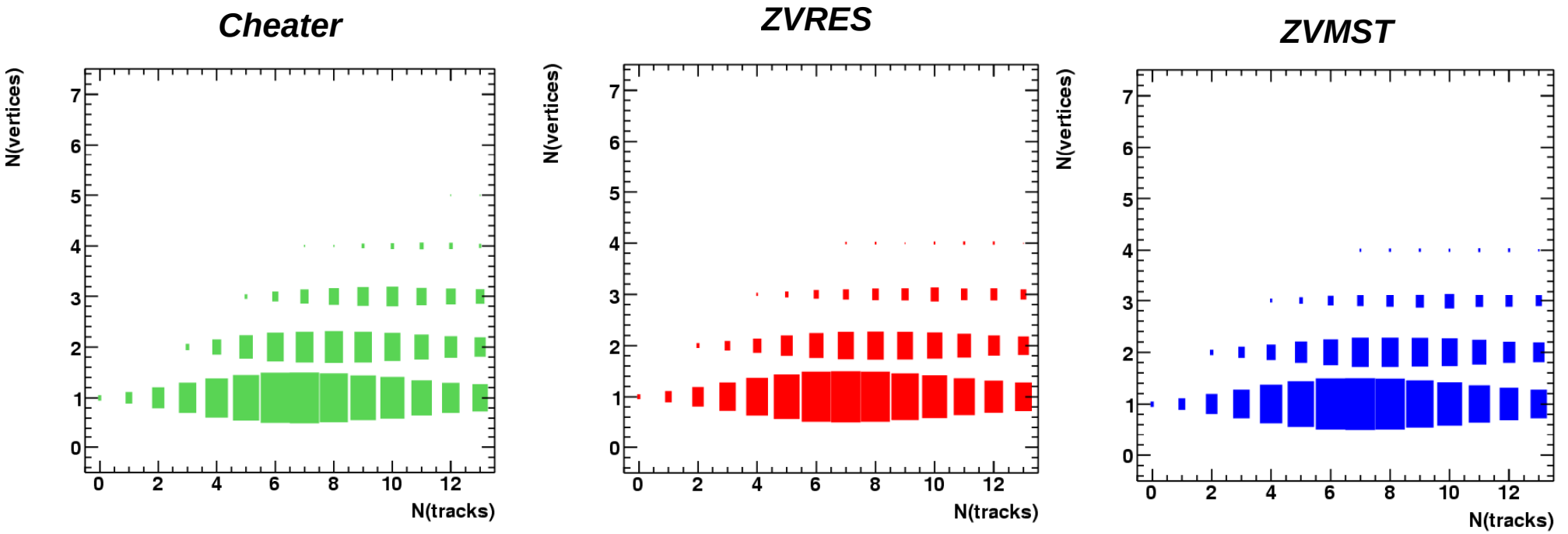
parameter	initial value	range investigated	step size	resulting default value
$d_{\min}\ [\mu m]$	300	$[50,\ 600]$	50	400
$f_{\min}$	0.0001	$[0.5 \cdot 10^{-4}, 1.5 \cdot 10^{-4}]$	$0.1 \cdot 10^{-4}$	0.0001
$\Delta f_{\max}$	0.9	$[0.85,\ 0.95]$	0.05	0.95
$\Delta V_{\min}$	0.1	$[0.05,\ 0.15]$	0.01	0.15

Multiplicity of edges and selected positions





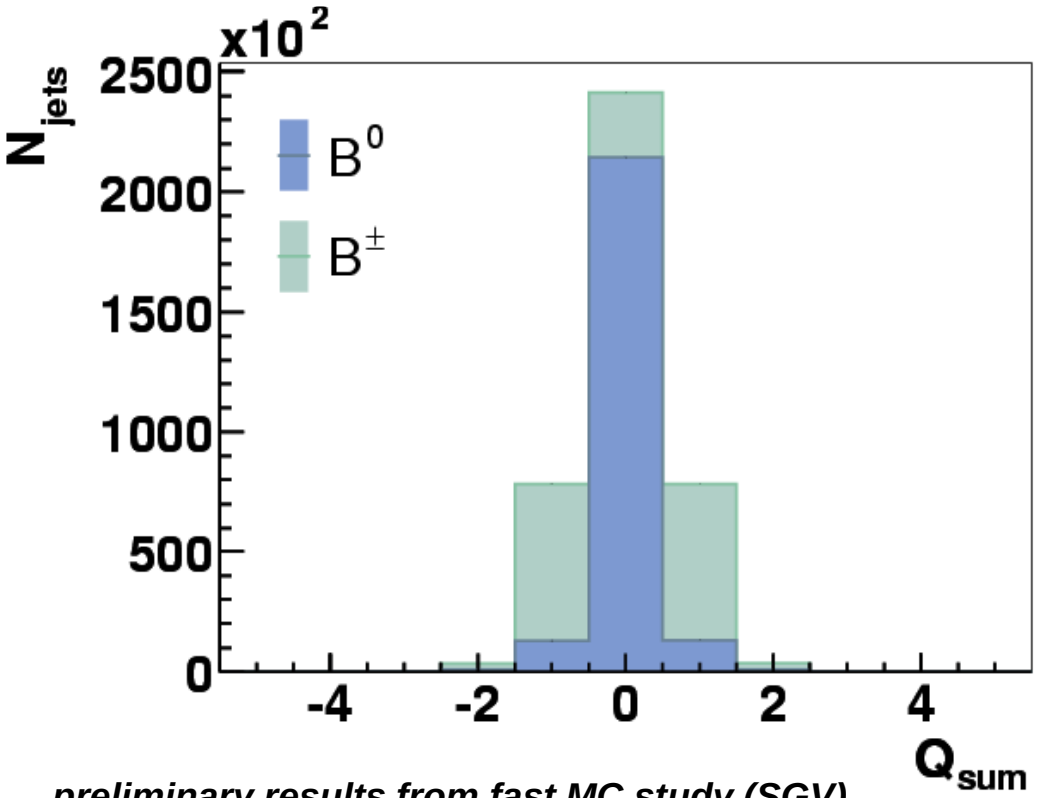
Vertex multiplicity vs input track multiplicity



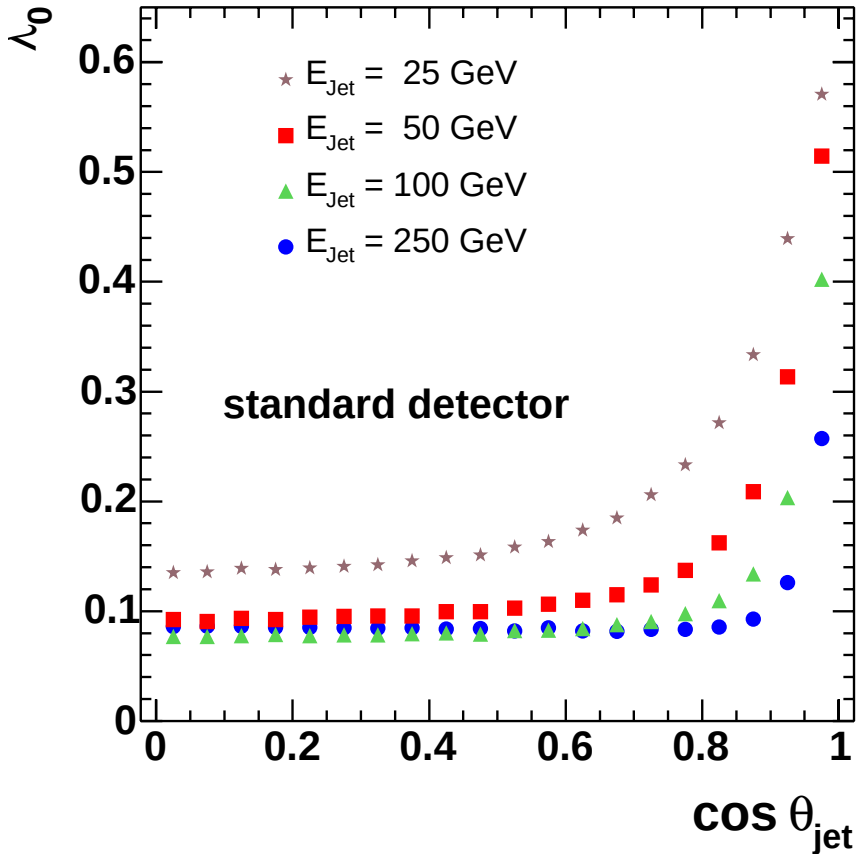
Performance of charge reconstruction: leakage rates

- define leakage rate λ_0 as probability of reconstructing neutral hadron as charged
- performance strongly depends on low momentum tracks:

  largest sensitivity to detector design for low jet energy, large cos θ



preliminary results from fast MC study (SGV)



Benchmark Physics Studies

- Benchmark physics processes should be typical of ILC physics and sensitive to detector design.
- A Physics Benchmark Panel comprising ILC theorists and experimentalists has published a list of recommended processes that will form the baseline for the selection of processes to be studied in the LoI- and engineering design phases.
- Following processes were highlighted as most relevant by the experts (hep-ex/0603010):

0. Single $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$, π^0, $K^{\pm}$, K^0_S, γ, $0 < |\cos\theta| < 1$, $0 < p < 500$ GeV

particularly sensitive to vertex detector design →

1. $e^+e^- \to f\bar{f}$, $f = e, \tau, u, s, c, b$ at $\sqrt{s}$=0.091, 0.35, 0.5 and 1.0 TeV;
2. $e^+e^- \to Z^0h^0 \to \ell^+\ell^- X$, $M_h = 120$ GeV at $\sqrt{s}$=0.35 TeV;
3. $e^+e^- \to Z^0h^0$, $h^0 \to c\bar{c}, \tau^+\tau^-, WW^*$, $M_h = 120$ GeV at $\sqrt{s}$=0.35 TeV;
4. $e^+e^- \to Z^0h^0h^0$, $M_h = 120$ GeV at $\sqrt{s}$=0.5 TeV;
5. $e^+e^- \to \tilde{e}^+_R\tilde{e}^-_R$ at Point 1 at $\sqrt{s}$=0.5 TeV;
6. $e^+e^- \to \tilde{\tau}^+_1\tilde{\tau}^-_1$, at Point 3 at $\sqrt{s}$=0.5 TeV;
7. $e^+e^- \to \tilde{\chi}^+_1\tilde{\chi}^-_1 / \tilde{\chi}^0_2\tilde{\chi}^0_2$ at Point 5 at $\sqrt{s}$=0.5 TeV;