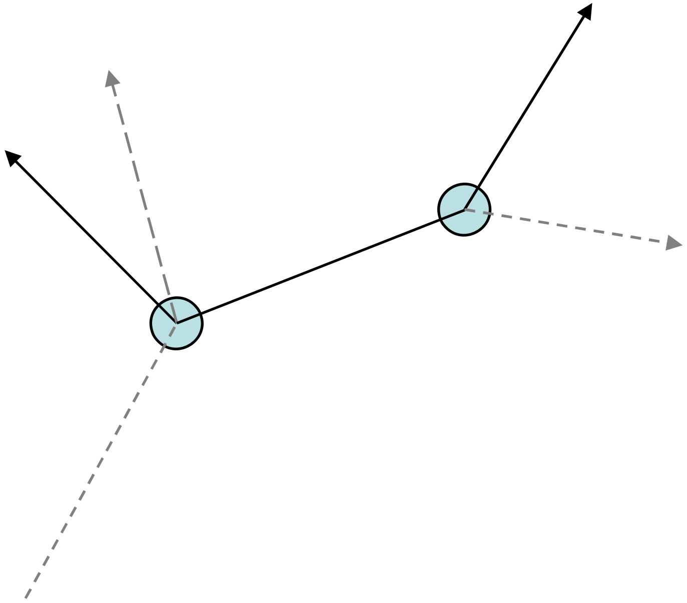
LCFI Vertex Package – Technical Summary

Ben Jeffery, Erik Devetak, Mark Grimes

(Oxford University) *(Bristol University)*



Package content

Nine new Marlin processors developed for the LCFI Vertex Package:

- **TrueJetFlavour** - Determine MC flavour of reconstructed jets
- **RPCut** - Select ReconstructedParticles based on Track parameters, number of hits etc.
- **PerEventIPFitter** - Find the event IP
- **ZVTOPZVRES** - Find secondary vertices topologically
- **ZVTOPZVKIN** - Find secondary vertices kinematically
- **FlavourTagInputs** - From vertices and tracks calculate discriminating variables for the neural net
- **NeuralNetTrainer** - Train networks
- **FlavourTag** - Calculate network output (tag value)
- **Plot** - Calculate purity and efficiency for the tags

Typical Run

Select Tracks

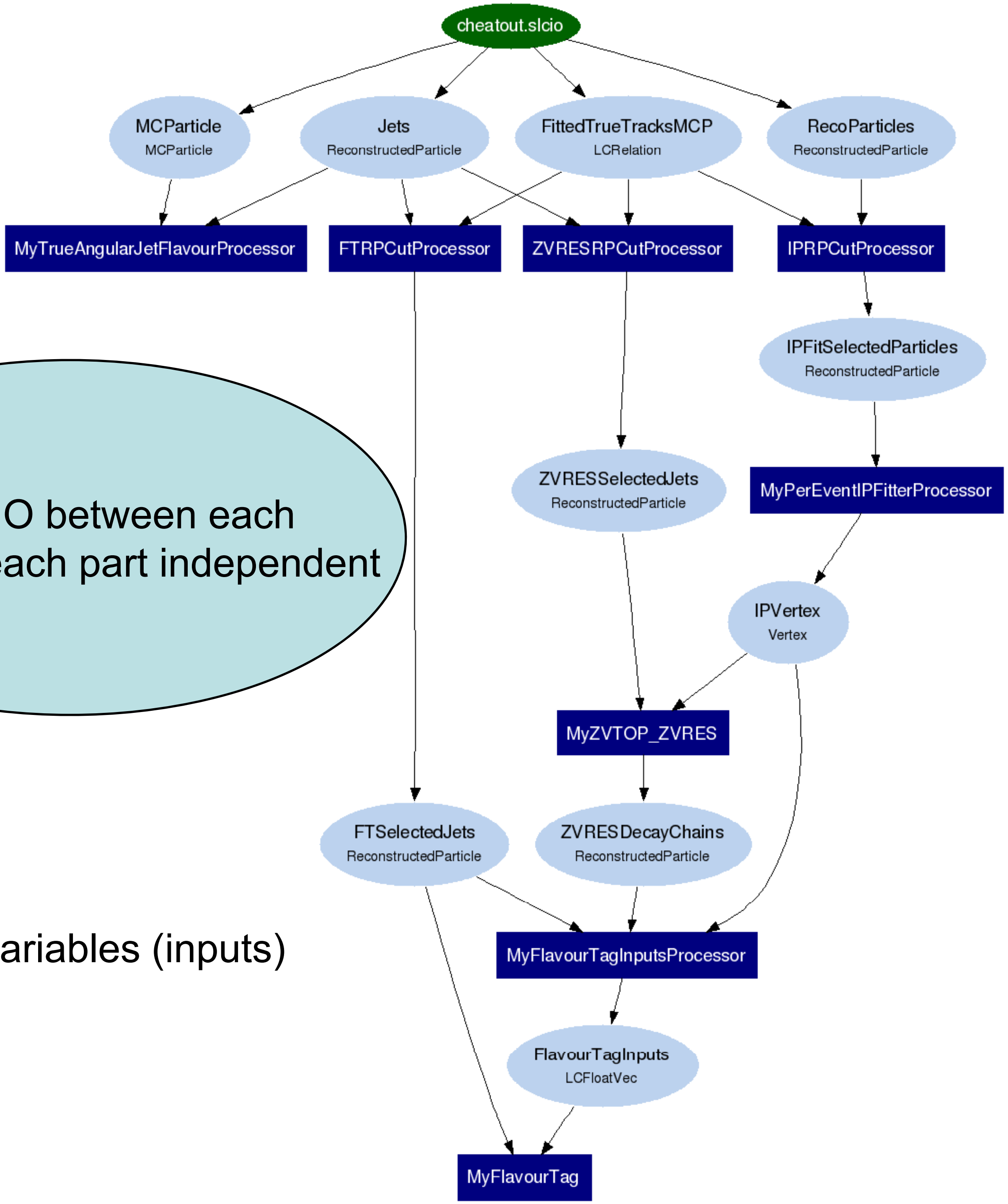
Fit IP

Vertex

Calculate discriminating variables (inputs)

Calculate network output

Pure LCIO between each processor – each part independent



Example Steering

- Example steering files are distributed with the code. (LCFIVertex/steering)
- One for each part of the tagging process – could easily be combined to single steering file.
- Sensible default parameters from previous Brahms study. Full study with Mokka input not yet performed.
- Also included is an example steering file for the rest of MarlinReco. (SimTrackerHits taken to ReconstructedParticle jets using the TrackCheater) as an example of how to obtain suitable input.

Order of example steering files:

cheattracks+jetfind.xml
truejetflavour.xml
ipfit.xml
zvres.xml
fti.xml
ft.xml

Total Time ~ 1s per jet
(2.5GHz p4)

Installation issues, etc.

- Available now as tar archive (cvs soon) from:
  http://www-flc.desy.de/ilcsoft/ilcsoftware/LCFIVertex/
- Installs as standalone package in $MARLINWORKDIR/packages.
- No external dependencies. (uses boost library but this is included internally)
- Requires latest version of MarlinReco (v00-03) if using TrackCheater for covariance matrices.
- Only tested under SL3 so far.
- Order ~5 gain in runtime without MARLINDEBUG.
- Currently **geometry independent** (no gearfile needed).

- Although default procedure (example steering) is tested this is a first release.
- Welcome comments/patches/bug reports – use the forum

- For more information than we can fit in here see the documentation included with the code. (instructions in README)
- A documentation update will be included in the next release.

RPCutProcessor

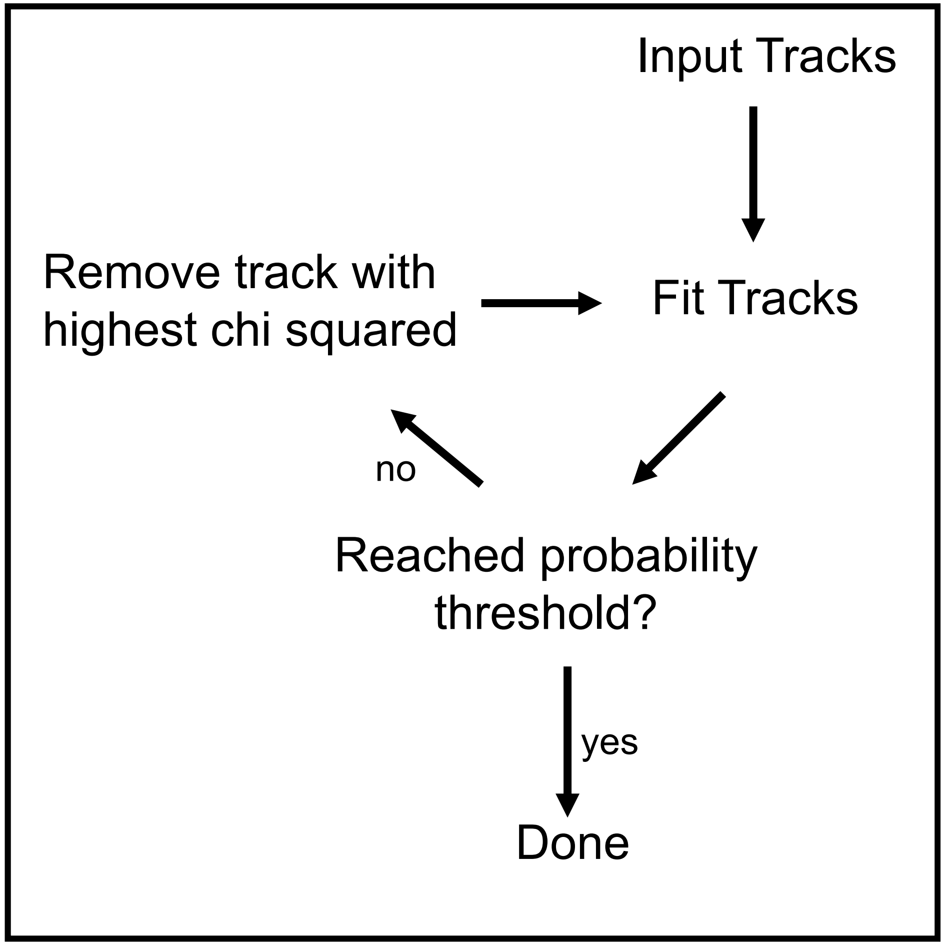
- Cuts ReconstructedParticles either directly from a collection or from those pointed to by another ReconstructedParticle (ie from a Jet)
- Either cuts from existing collection or creates a new collection
- As the rest of the package only works on ReconstructedParticles with a geometric Track representation, those without are cut.
  (ie. ReconstructedParticle::getTracks().empty() == true)

- Input
  - Set of ReconstructedParticles
  - (eg Jets from SatoruJetFinder)
- Output
  - Set of ReconstructedParticles
- Key Parameters
  - SubParticleLists – Cut from an RP not a collection
  - WriteNewCollection – Leave input untouched
  - Cut Values

Used here to clean tracks for IP Fitting,
Vertexing and Flavour Tag calculation,
but can be used for any other purpose.

IP Fitter

- Input
  - Set of ReconstructedParticles (eg From RPCut)
- Output
  - Vertex
- Key Parameters
  - Vertex Probability Threshold
  - Default IP (If no vertex is found then the default is used.)



Input Tracks
Fit Tracks
Reached probability threshold?
no
Remove track with highest chi squared
yes
Done

This processor was developed as a short term “place holder” as this information was needed for the tag, but was not available in the Marlin/LCIO framework.

The ZVTOP vertex fitter was recycled, the processor should be considered sub-optimal until something more clever is implemented – ideally that used at SLD (average over events in xy plane)

ZVTOP-ZVRES & ZVKIN Processor

- Input
  - ReconstructedParticles - Jets
  - *(optional)* Vertex – IP
- Output
  - ReconstructedParticles
    DecayChains
  - ReconstructedParticles
    DecayChainTracks
  - Vertices
- Key Parameters
  - Manual IP
  - Output Chi Squared
  (Chi squared of tracks in vertices)
  - *Algorithmic parameters*

Both processors find set of vertices in jets:

ZVRES – Topological
(D. Jackson, NIM A 388 (1997) 247)

ZVKIN – Kinematic
(Described in appendix to J. Thom SLAC-R-585, Jan 2002)

Tracks need to be filtered for quality previously (eg by RPCut) and must have well-formed, accurate covariance matrices.

ZVRES previously studied with Brahms, also tested in this framework. *(See Ben Jeffery's talk at the Valencia '06 workshop)*

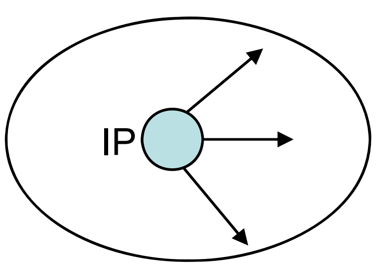
ZVKIN is highly experimental.

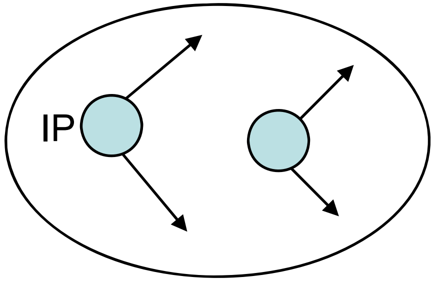
The output for both algorithms is identical and always contains at least one vertex (the IP).

The output is in the form of decay chains:

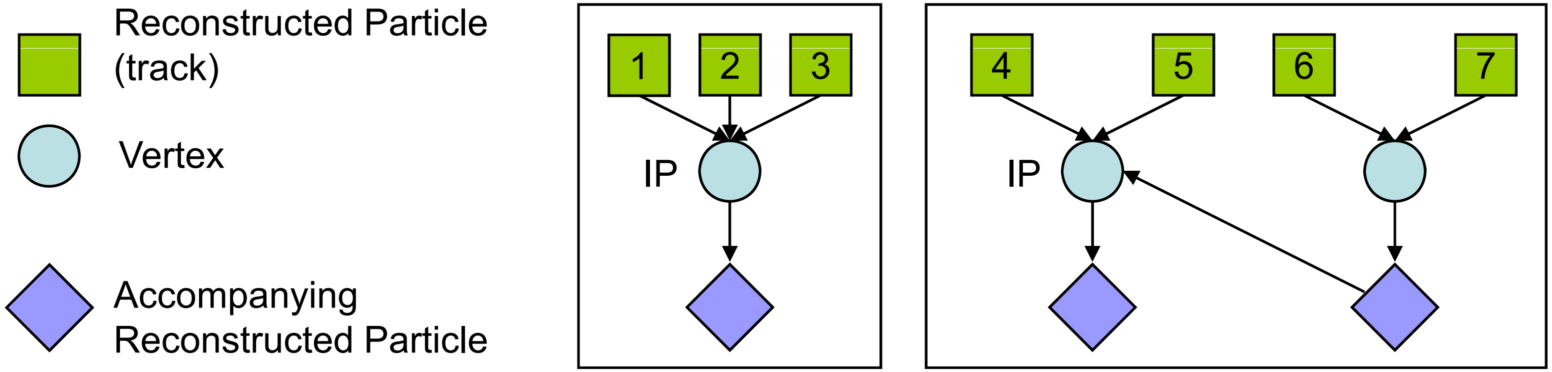
Storage Of Vertexing Result

- ZVTOP Produces "Decay Chains" – sets of one or more vertices eg:

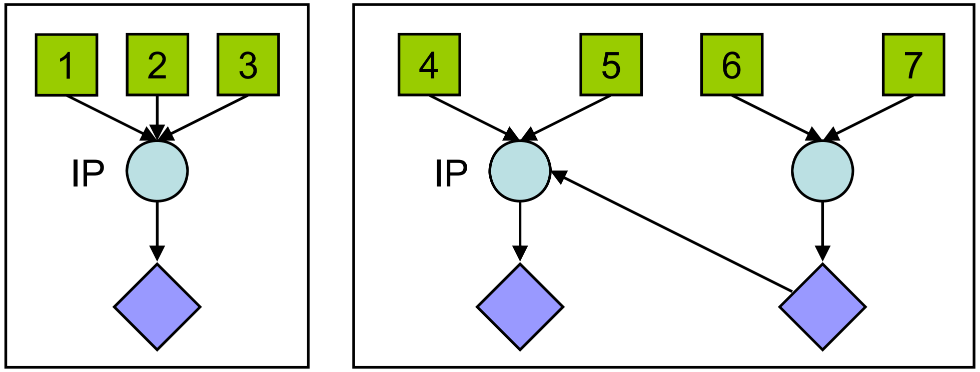




In LCIO each Vertex has an accompanying ReconstructedParticle which represents the decaying particle. This holds kinematic information. Each ReconstructedParticle points to its "startVertex".


Reconstructed Particle (track)
Vertex
Accompanying Reconstructed Particle
1
2
3
IP
4
5
6
7
IP

Storage Of Vertexing Result



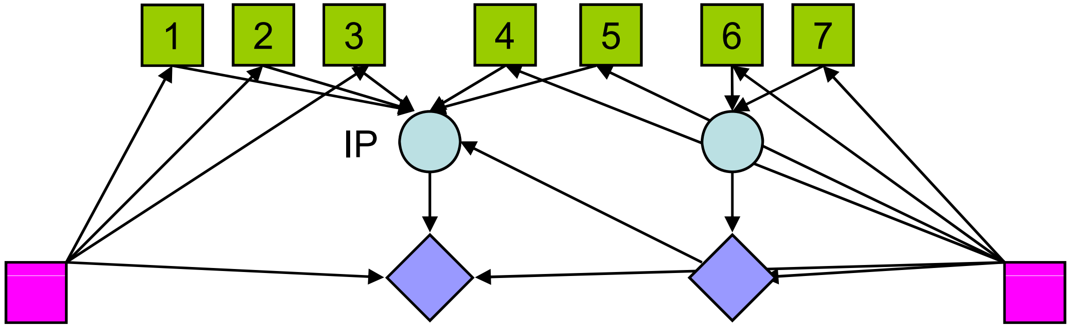
🟩 Reconstructed Particle (track)

● Vertex

◆ Accompanying Reconstructed Particle

🟪 Reconstructed Particle (decay chain)

If these are in the same event they share the IP Vertex:



ReconstructedParticle representing decay chain points to all tracks in that chain (one for each jet)

Result is three collections:

● Vertices

◆ 🟩 DecayChainTracks

🟪 DecayChains

Code examples of accessing this information are in the documentation