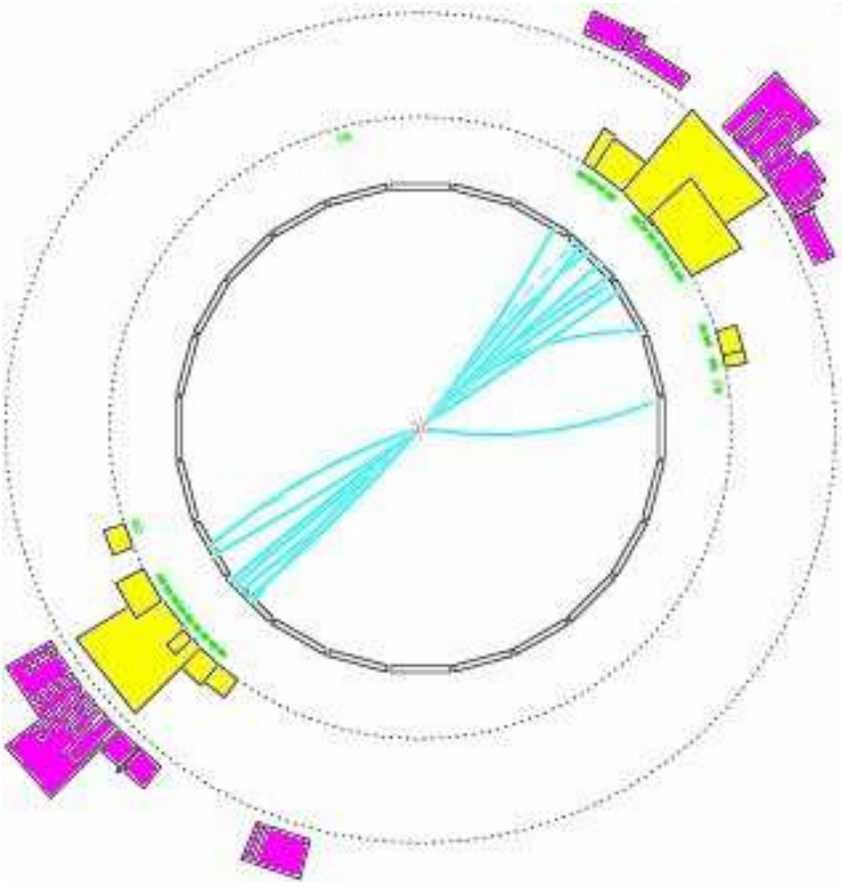
Jets

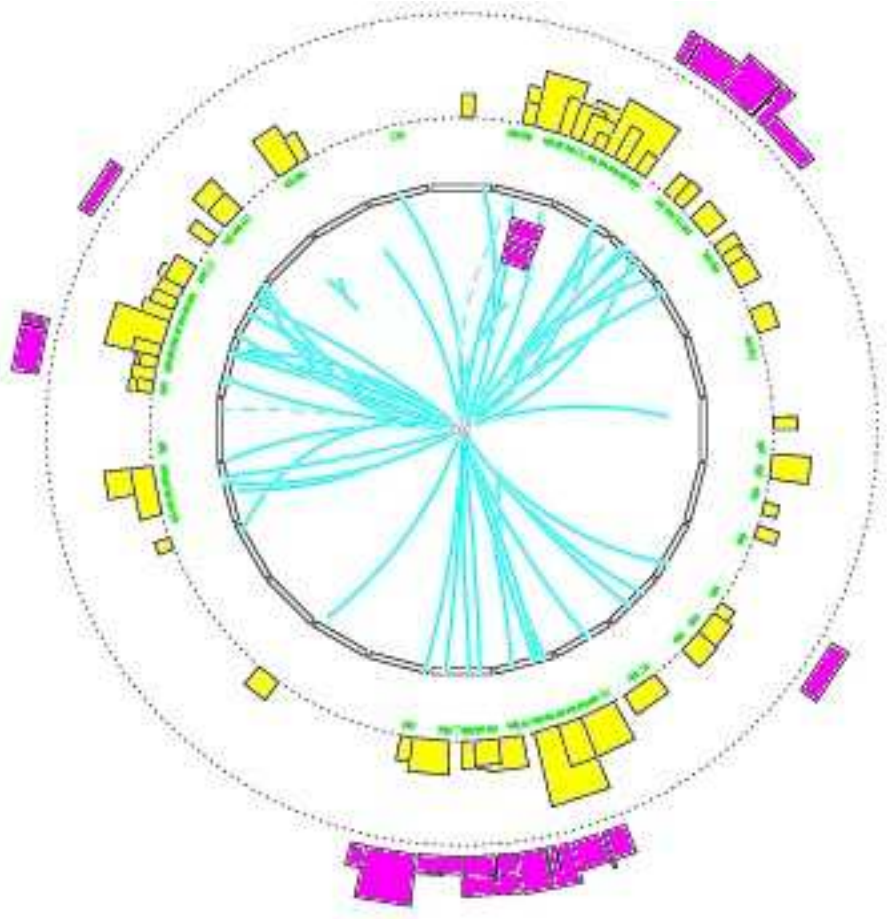
Gavin Salam

LPTHE, CNRS and UPMC (Univ. Paris 6)

CTEQ MCNet school
Debrecen, Hungary, August 2008

Jets





Jets are everywhere in QCD
Our *window on partons*

But *not* the same as partons:
Partons ill-defined; jets *well-definable*

Why do we see jets? Partons framgent

Perturbatively

- Quarks fragment: soft & collinear divergences for gluon emission
- Gluons fragment: soft & collinear divergences for gluon emission
  collinear divergences for quark emission
- Even perturbative coupling is not so small

Non-perturbatively

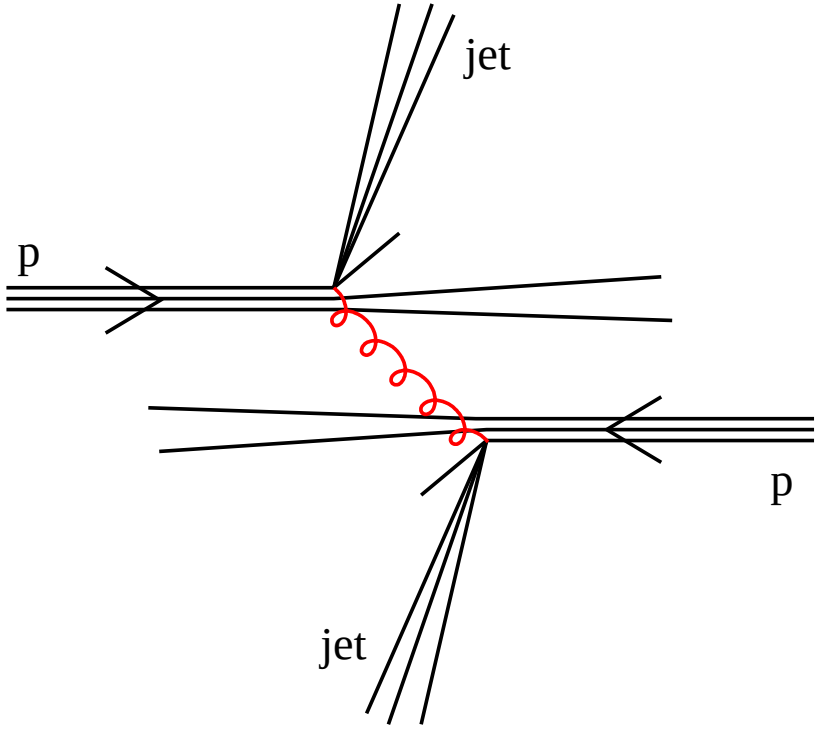
- precise process long way from being understood, even by lattice
- good models contain many parameters — complex process

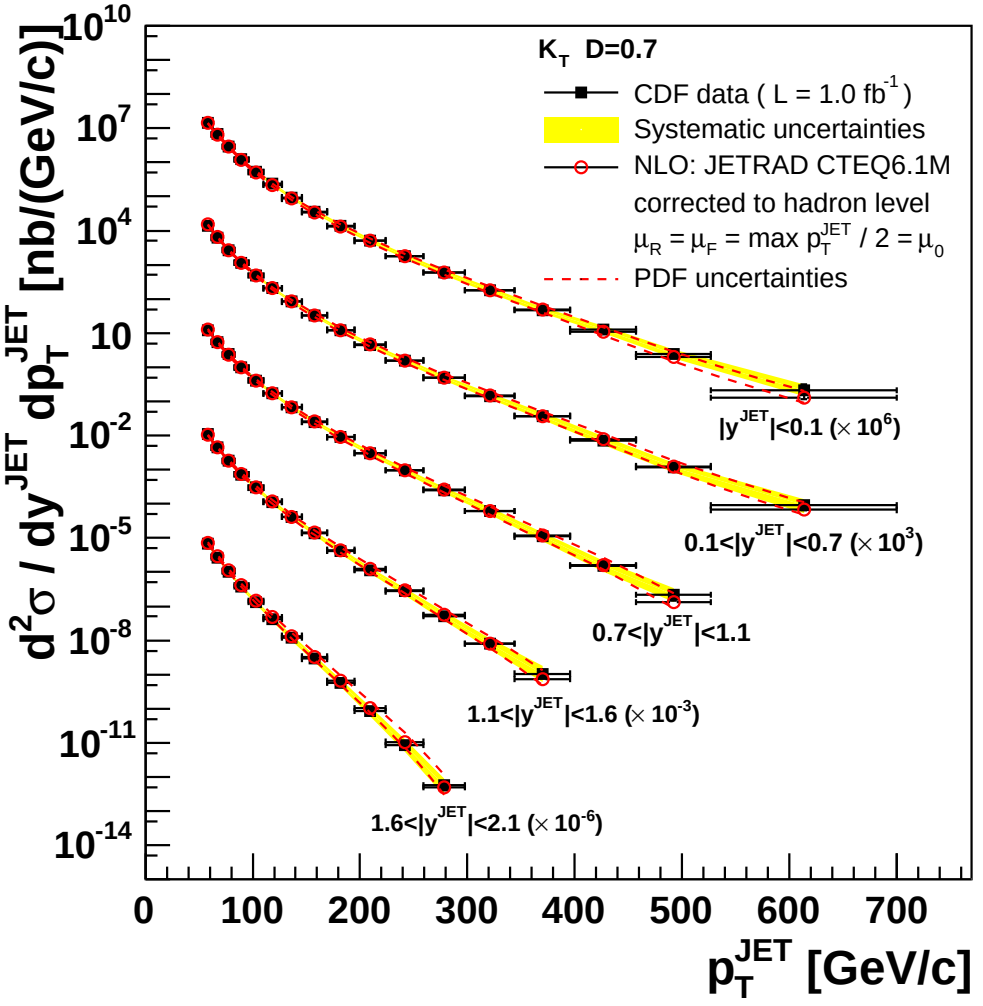
High-energy partons unavoidably lead to collimated bunches of hadrons.

See lectures by Dave Soper, Mike Seymour

Jets from scattering of partons

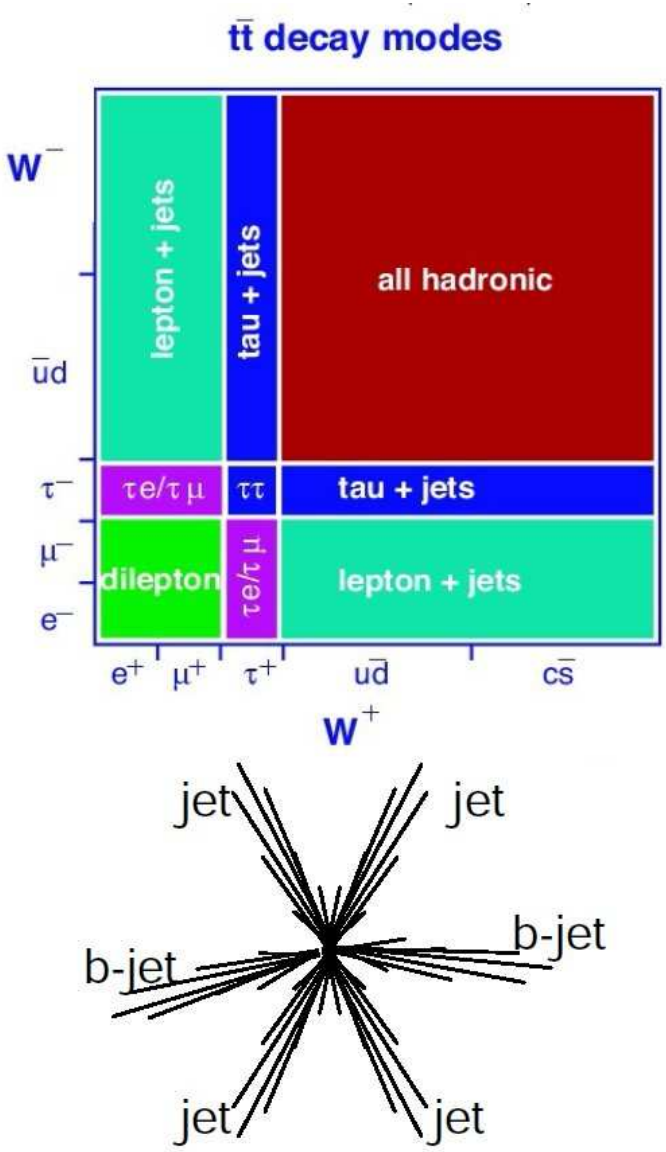
Jets are unavoidable at hadron colliders, e.g. from parton scattering





Jet cross section: data and theory agree over many orders of magnitude $\Leftrightarrow$ probe of underlying interaction

Jets from heavy decays



All-hadronic
(BR~46%, huge bckg)

picture: Juste LP05

Heavy objects: multi-jet final-states

- 10^7 $t\bar{t}$ pairs for 10 fb^{-1}
- Vast # of QCD multijet events

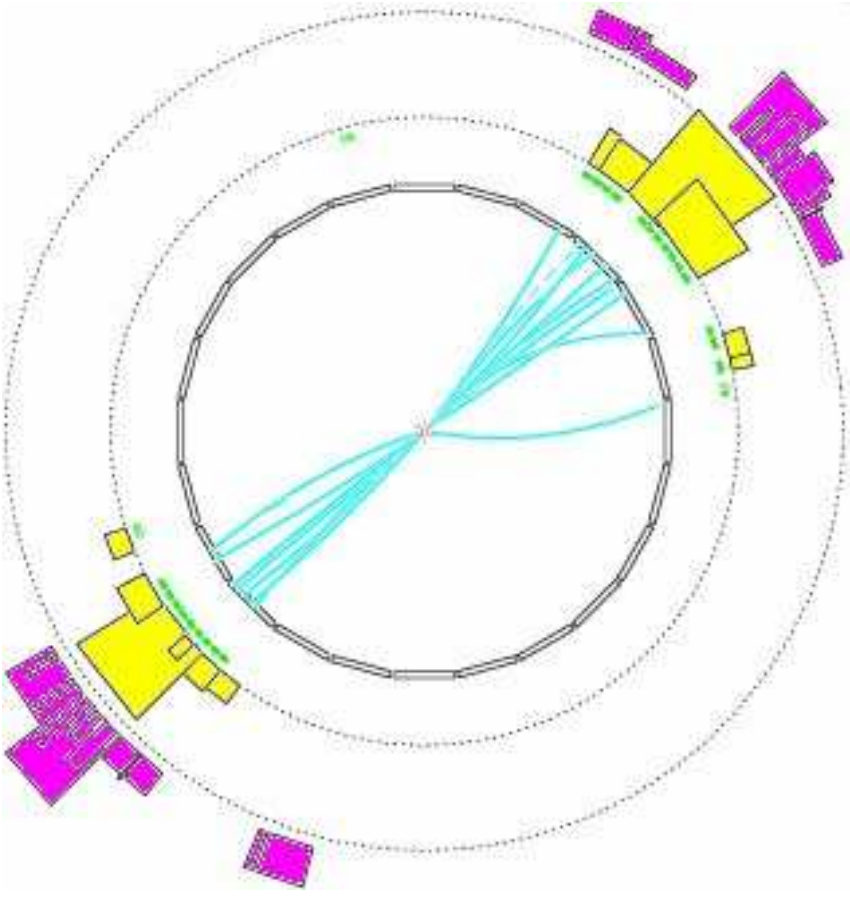
# jets	# events for 10 fb^{-1}
3	$9 \cdot 10^8$
4	$7 \cdot 10^7$
5	$6 \cdot 10^6$
6	$3 \cdot 10^5$
7	$2 \cdot 10^4$
8	$2 \cdot 10^3$

Tree level

$p_t(\text{jet}) > 60$ GeV, $\theta_{ij} > 30$ deg, $|y_{ij}| < 3$

Draggiotis, Kleiss & Papadopoulos '02

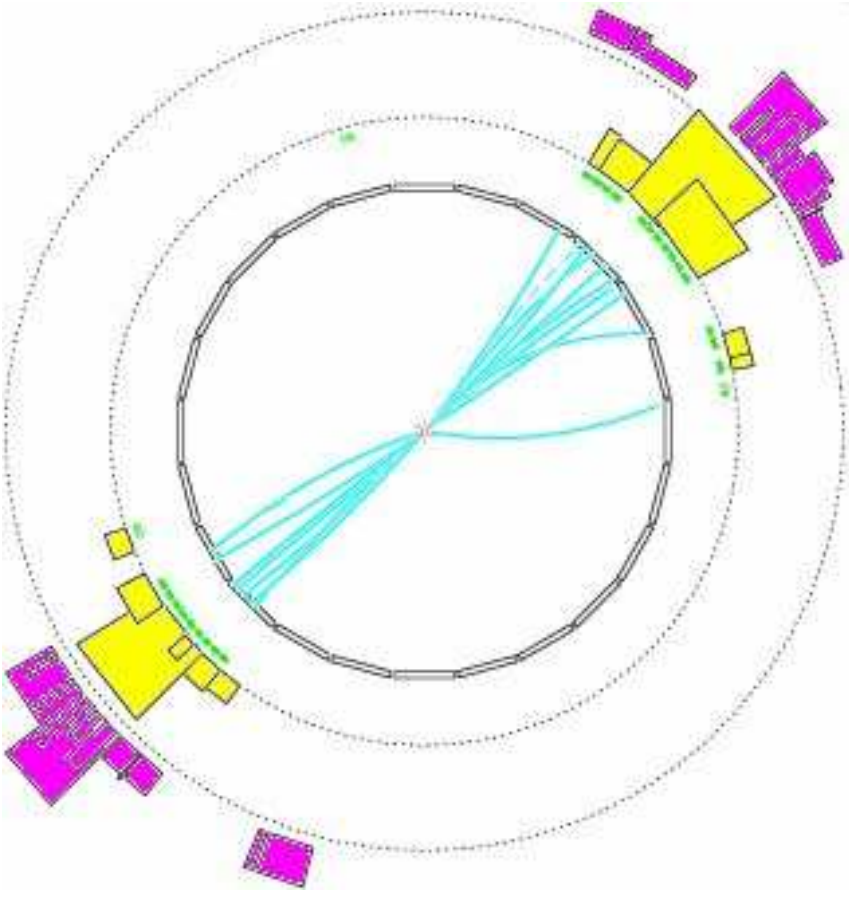
Seeing v. defining jets



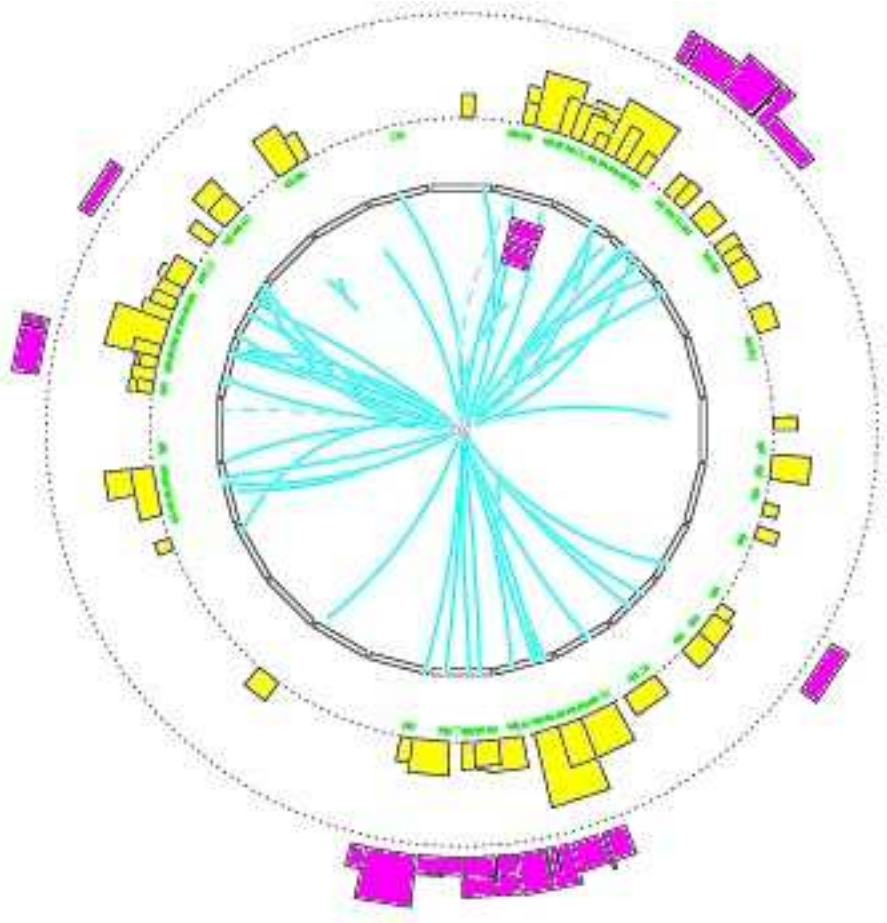
Jets are what we see.
Clearly(?) 2 jets here

How many jets do you see?
Do you really want to ask yourself this question for 10^8 events?

Seeing v. defining jets

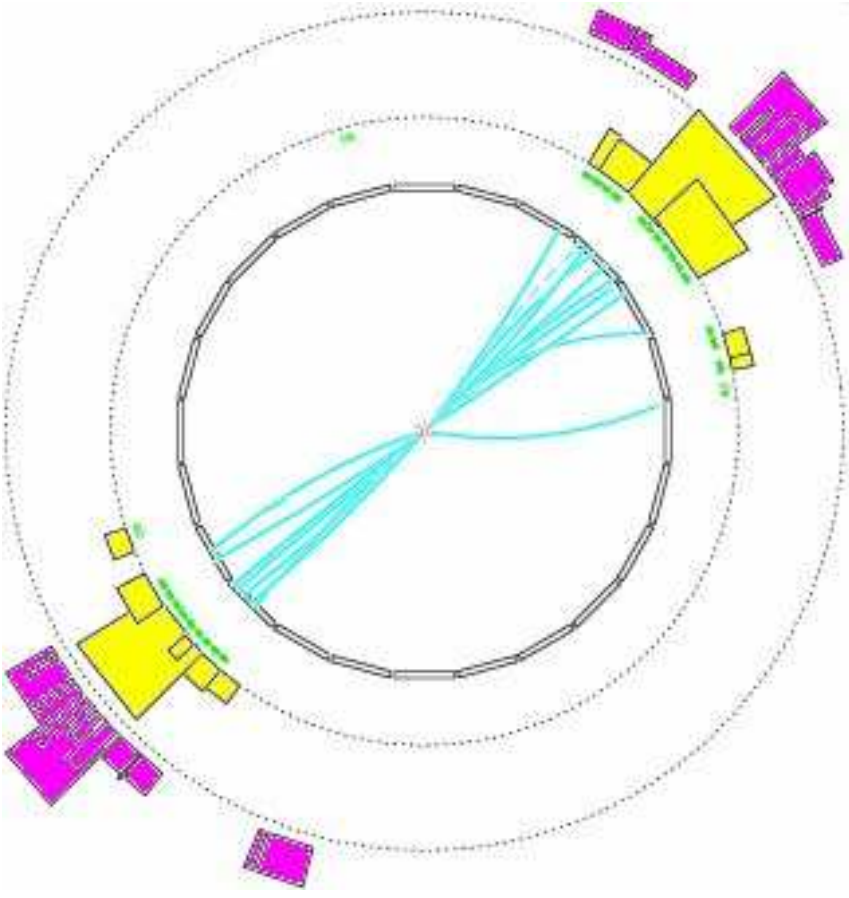


Jets are what we see.
Clearly(?) 2 jets here

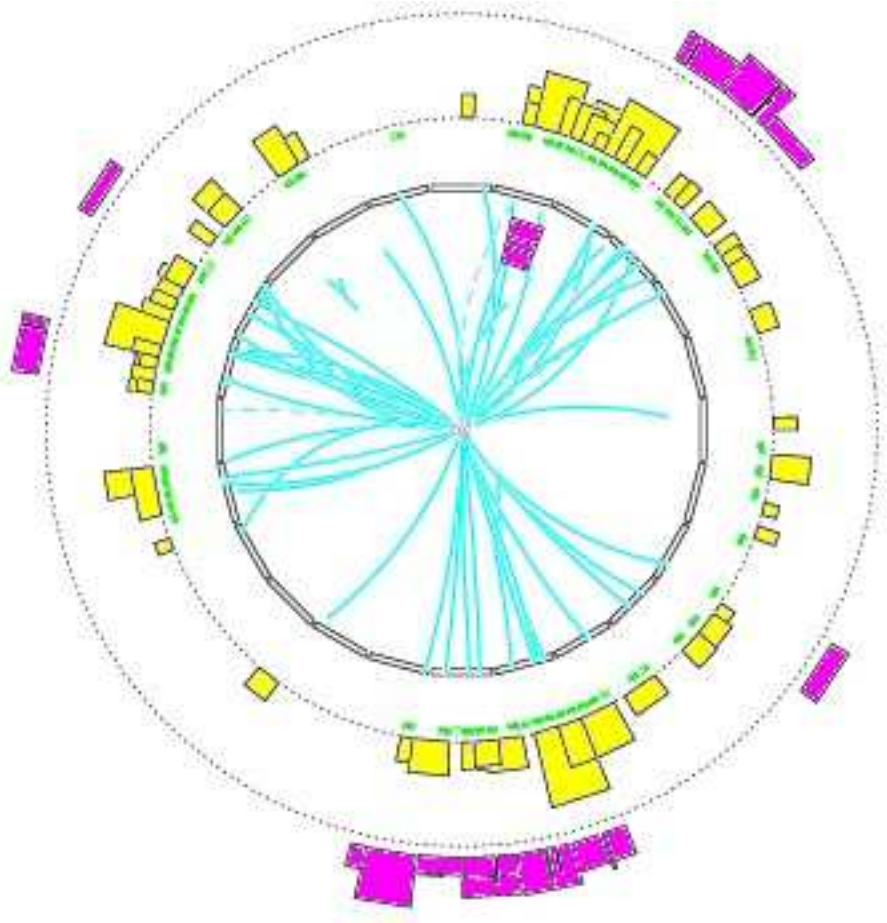


How many jets do you see?
Do you really want to ask yourself this question for 10^8 events?

Seeing v. defining jets



Jets are what we see.
Clearly(?) 2 jets here

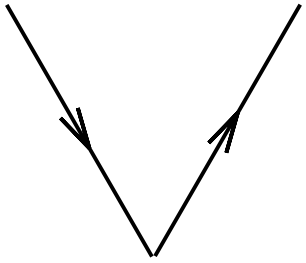


How many jets do you see?
Do you really want to ask yourself this question for 10^8 events?

Jet definitions

- *A jet definition is a fully specified set of rules for projecting information from 100's of hadrons, onto a handful of parton-like objects:*
  - or project 1000's of calorimeter towers
  - or project dozens of (showered) partons
  - or project a handful of (unshowered) partons
- Resulting objects (jets) used for many things, e.g. :
  - reconstructing decaying massive particles — e.g. top $\longrightarrow$ 3 jets
  - constraining proton structure
  - as a theoretical tool to attribute structure to an event
- You *lose much information* in projecting event onto jet-like structure:
  - Sometimes information you had no idea how to use
  - Sometimes information you may not trust, or of no relevance

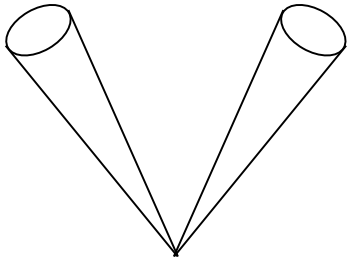
Jets as projections

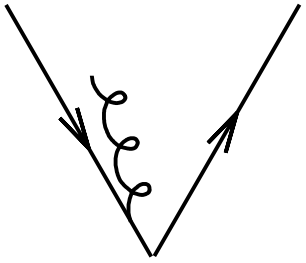


LO partons

Jet | Def^n

jet 1 jet 2

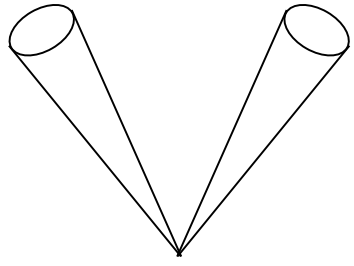


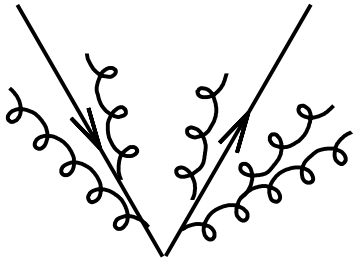


NLO partons

Jet | Def^n

jet 1 jet 2

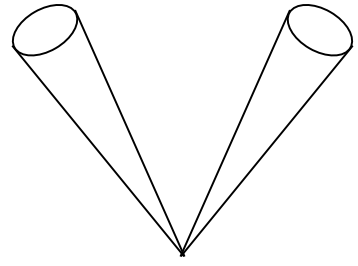


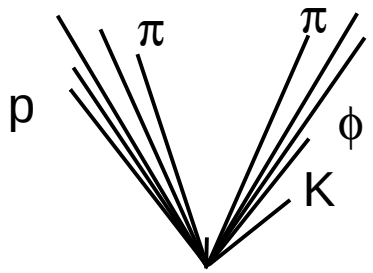


parton shower

Jet | Def^n

jet 1 jet 2

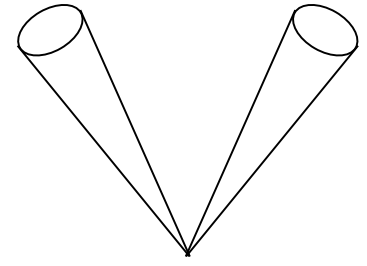




hadron level

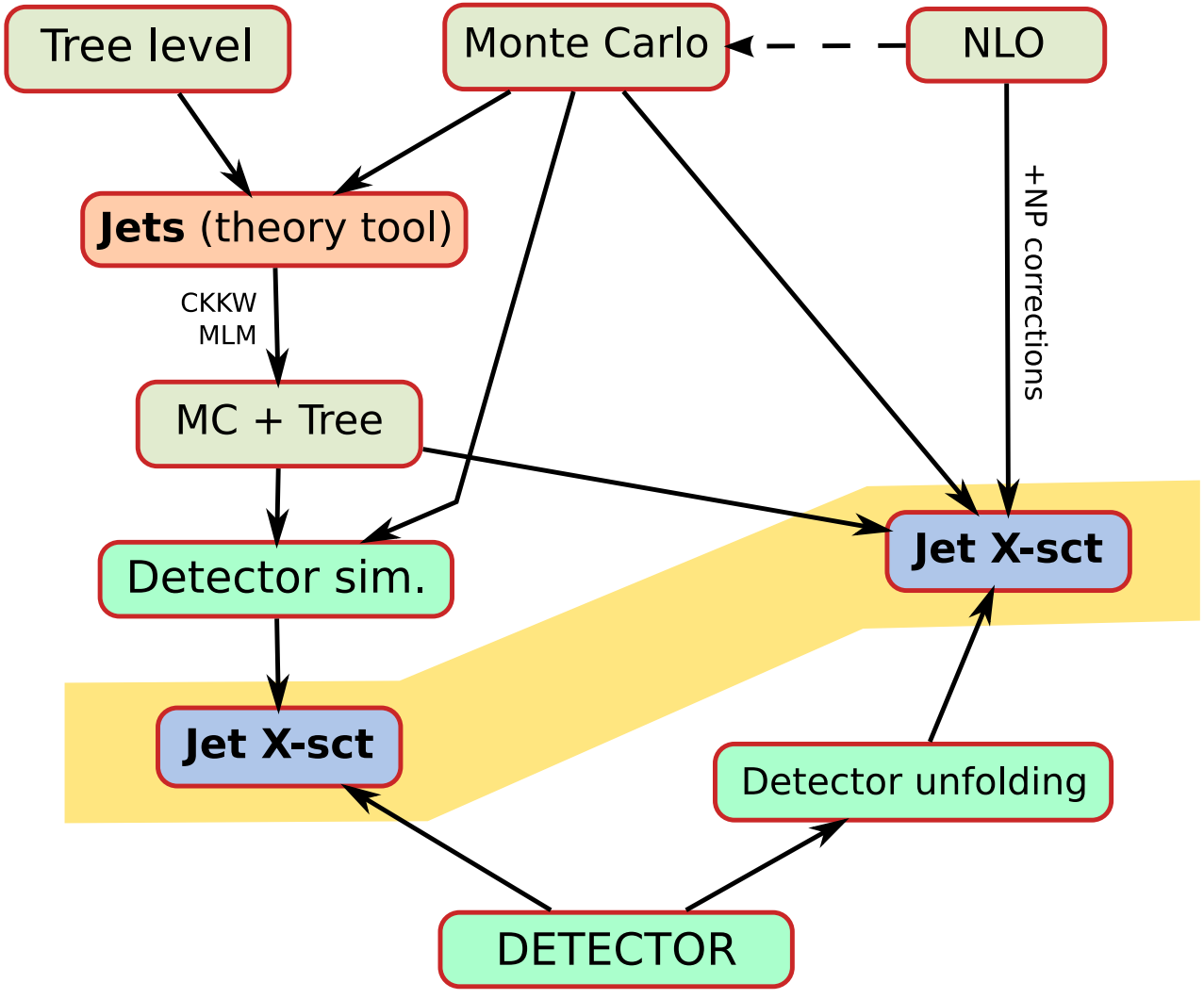
Jet | Def^n

jet 1 jet 2



Projection to jets should be resilient to QCD effects

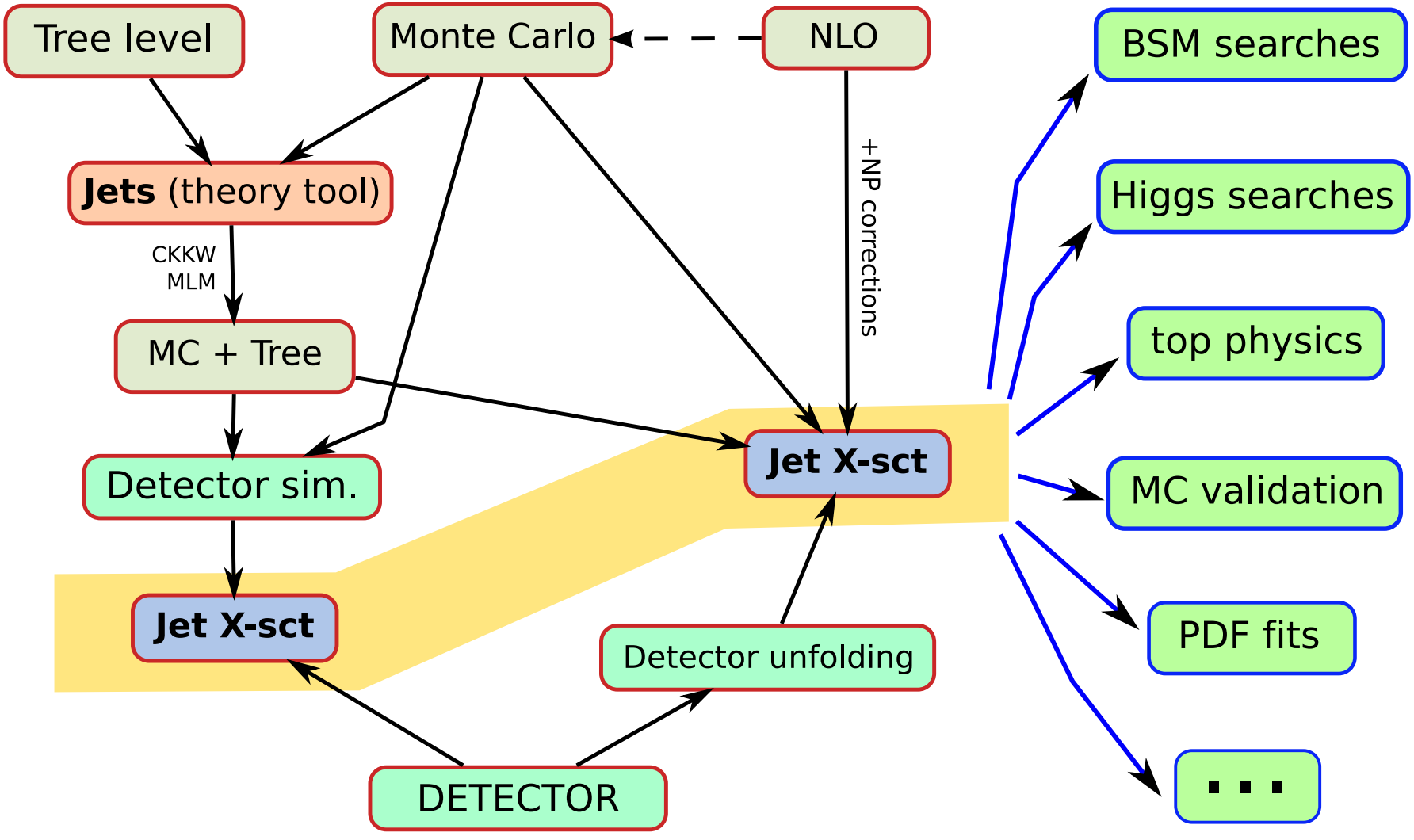
QCD jets flowchart



Jet (definitions) provide central link between expt., "theory" and theory

And jets are an input to almost all analyses

QCD jets flowchart



Jet (definitions) provide central link between expt., "theory" and theory

And jets are an input to almost all analyses

These lectures

Aims: to provide you with

- the "basics" needed to understand what goes into current jet-based measurements;
- some insight into the issues that are relevant when thinking about a jet measurement

Structure:

- General considerations
- Common jet definitions — we'll look at 2 broad classes
  - Sequential recombination — today
  - Cone — today & tomorrow
- The physics of jets [briefly] — tomorrow

There is no unique jet definition

The construction of a jet is unavoidably ambiguous. On at least two fronts:

1. which particles get put together into a common jet? — Jet algorithm + parameters
2. how do you combine their momenta? — Recombination scheme

Most commonly used: direct 4-vector sums (E-scheme)

Taken together, these different elements specify a choice of jet definition

There is no unique jet definition

The construction of a jet is unavoidably ambiguous. On at least two fronts:

1. which particles get put together into a common jet? — Jet algorithm + parameters
2. how do you combine their momenta? — Recombination scheme

Most commonly used: direct 4-vector sums (E-scheme)

Taken together, these different elements specify a choice of jet definition

The power of ambiguity

- Physical results (particle discovery, masses, PDFs, coupling) should be independent of your choice of jet definition

  a bit like renormalisation scale/scheme invariance

  Tests independence on modelling of radiation, hadronisation, etc.

- Except when there is a good reason for this not to be the case

Jetography, like photography

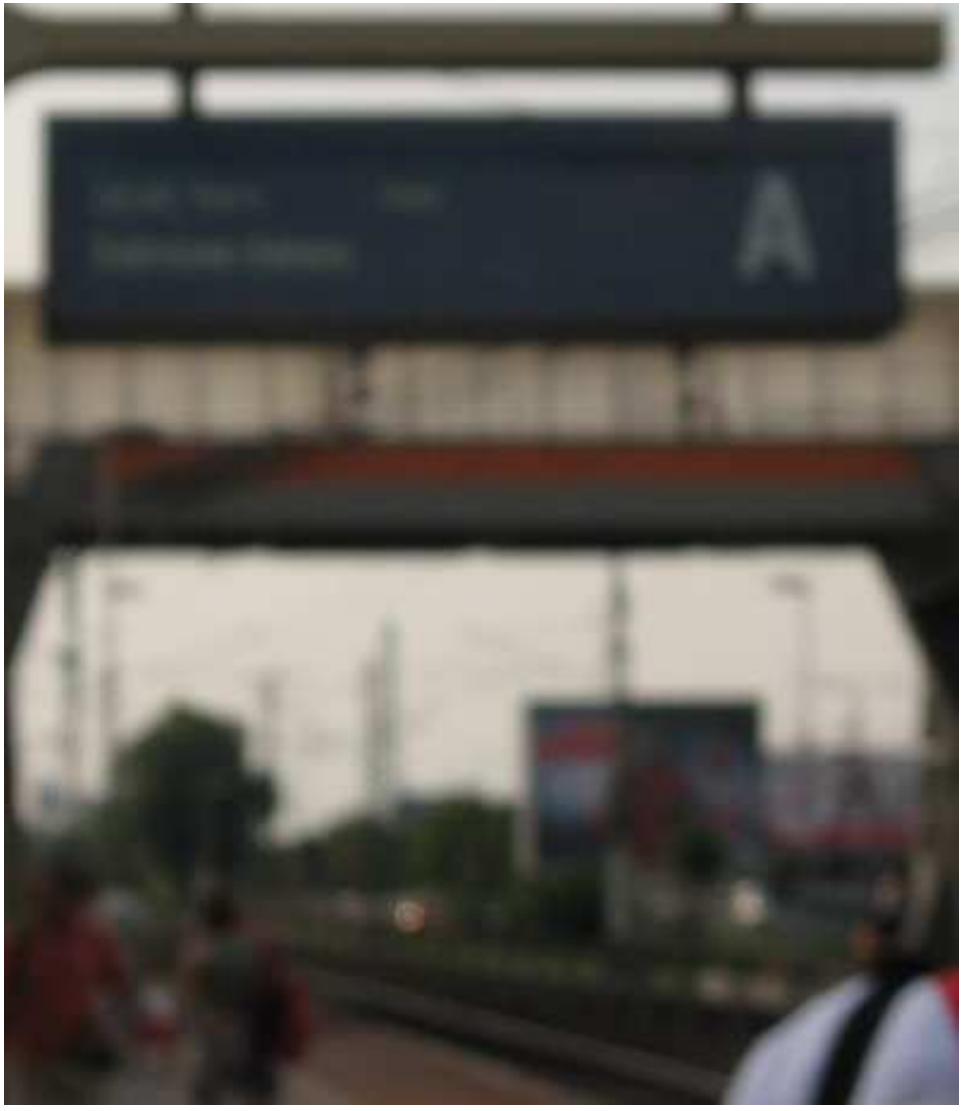
- Fine detail on bus ticket to train station — shoot from close up, focus = 40cm

[get to train station]

- Keep focus at 40cm
- Reset focus to 6m

Catch correct train

Jetography, like photography



- Fine detail on bus ticket to train station — shoot from close up, focus = 40cm

[get to train station]

- Keep focus at 40cm
- Reset focus to 6m

Catch correct train

Jetography, like photography

- Fine detail on bus ticket to train station — shoot from close up, focus = 40cm

[get to train station]

- Keep focus at 40cm
- Reset focus to 6m

Catch correct train

Jetography, like photography

- Fine detail on bus ticket to train station — shoot from close up, focus = 40cm

[get to train station]

- Keep focus at 40cm
- Reset focus to 6m

Catch correct train

Not all ambiguity is allowed

Jets should be **invariant** with respect to certain modifications of the event:

- collinear splitting
- infrared emission

Why?

- Because otherwise lose real-virtual cancellation in NLO/NNLO QCD calculations $\longrightarrow$ divergent results
- Hadron-level 'jets' fundamentally non-perturbative
- Detectors resolve neither full collinear nor full infrared event structure

Known as **infrared and collinear safety**

Not all ambiguity is allowed

Jets should be **invariant** with respect to certain modifications of the event:

- collinear splitting
- infrared emission

Why?

- Because otherwise lose real-virtual cancellation in NLO/NNLO QCD calculations $\longrightarrow$ divergent results
- Hadron-level 'jets' fundamentally non-perturbative
- Detectors resolve neither full collinear nor full infrared event structure

Known as **infrared and collinear safety**

Not all ambiguity is allowed

Jets should be **invariant** with respect to certain modifications of the event:

- collinear splitting
- infrared emission

Why?

- Because otherwise lose real-virtual cancellation in NLO/NNLO QCD calculations $\longrightarrow$ divergent results
- Hadron-level 'jets' fundamentally non-perturbative
- Detectors resolve neither full collinear nor full infrared event structure

Known as **infrared and collinear safety**

Two main classes of jet alg.

Sequential recombination (k_t, etc.)

- bottom-up
- successively undoes QCD branching

Cone

- top-down
- centred around idea of an 'invariant', directed energy flow

Sequential recombination jet algorithms

k_t/Durham algorithm

Majority of QCD branching is soft & collinear, with following divergences:

$$[dk_j]|M^2_{g\to g_i g_j}(k_j)| \simeq \frac{2\alpha_s C_A}{\pi}\frac{dE_j}{\min(E_i,E_j)}\frac{d\theta_{ij}}{\theta_{ij}}\,, \qquad (E_j \ll E_i\,,\ \theta_{ij} \ll 1)\,.$$

To invert branching process, take pair with strongest divergence between them — they're the most *likely* to belong together.

This is basis of **k_t/Durham algorithm** (e^+e^-):

1. Calculate (or update) distances between all particles i and j:

$$y_{ij} = \frac{2\min(E_i^2,E_j^2)(1-\cos\theta_{ij})}{Q^2}$$

NB: relative k_t between particles

2. Find smallest of y_{ij}
   - If $> y_{cut}$, stop clustering
   - Otherwise recombine i and j, and repeat from step 1

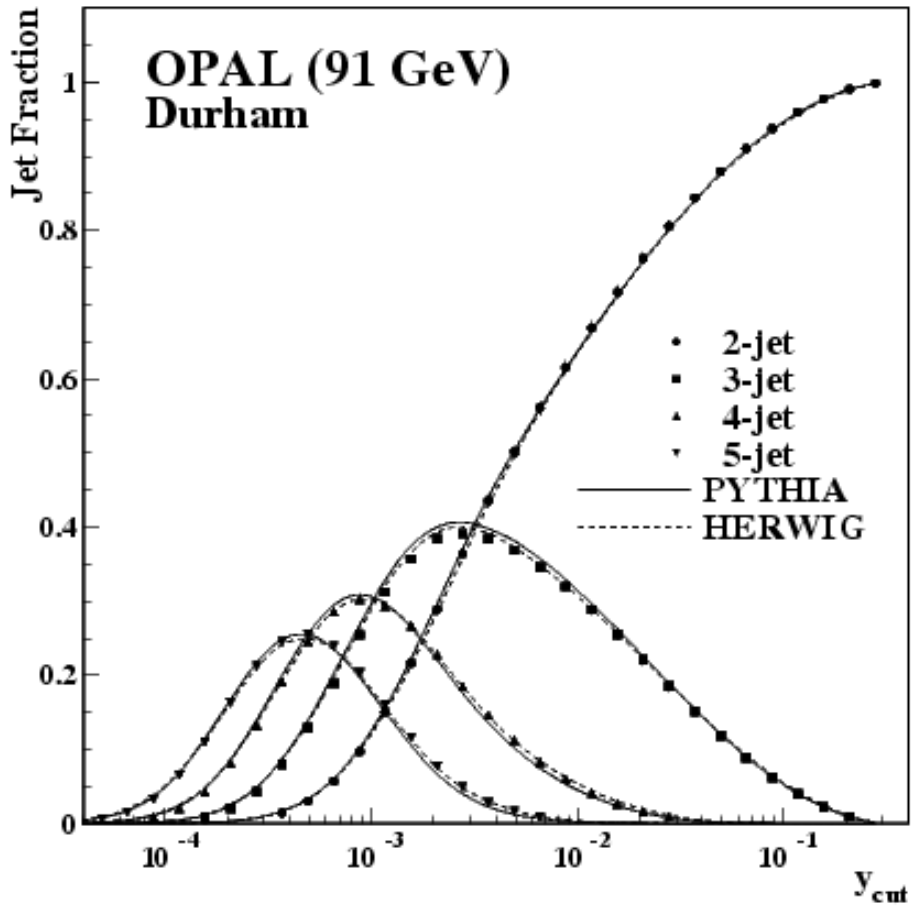
Catani, Dokshitzer, Olsson, Turnock & Webber '91

k_t/Durham algorithm features

- Gives hierarchy to event and jets

  Event can be specified by y_{23}, y_{34}, y_{45}.

- Resolution parameter related to minimal transverse momentum between jets



Most widely-used jet algorithm in e^+e^-

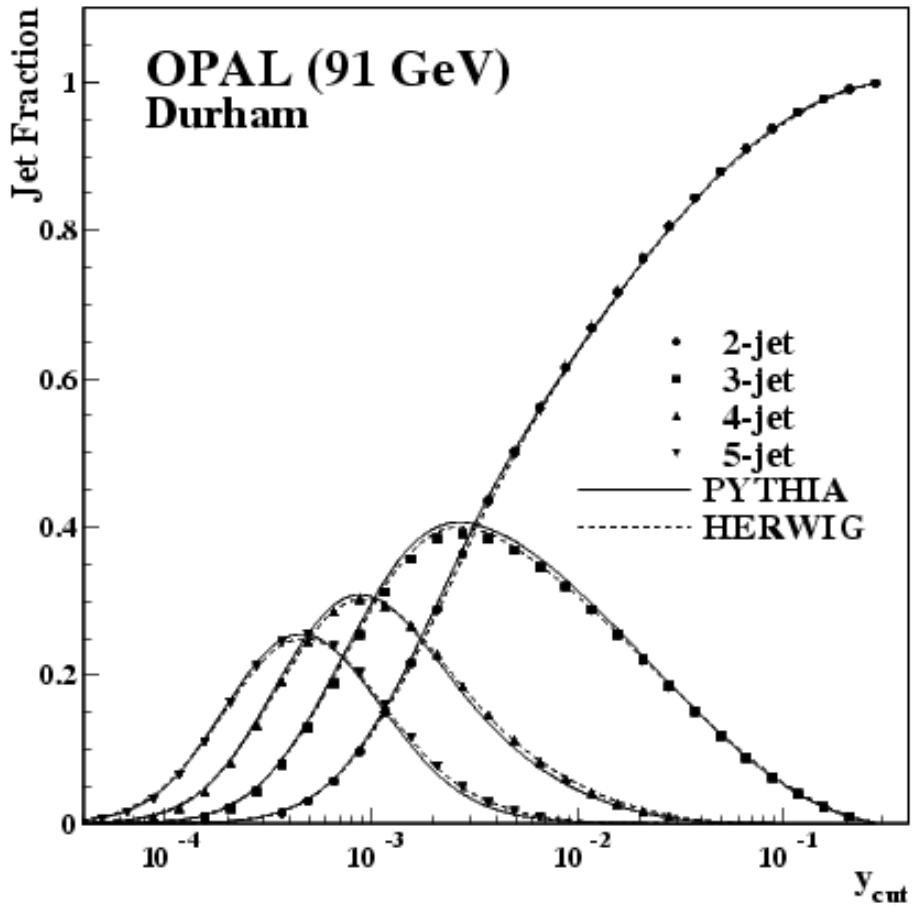
- Collinear safe: collinear particles recombined early on
- Infrared safe: soft particles have no impact on rest of clustering seq.

k_t/Durham algorithm features

- Gives hierarchy to event and jets

  Event can be specified by y_{23}, y_{34}, y_{45}.

- Resolution parameter related to minimal transverse momentum between jets



Most widely-used jet algorithm in e^+e^-

- Collinear safe: collinear particles recombined early on
- Infrared safe: soft particles have no impact on rest of clustering seq.

k_t alg. at hadron colliders

1st attempt

- Lose absolute normalisation scale Q. So use unnormalised d_ij rather than y_{ij}:

$$d_{ij} = 2\min(E_i^2, E_j^2)(1 - \cos\theta_{ij})$$

- Now also have *beam remnants* (go down beam-pipe, not measured) Account for this with particle-beam distance

$$d_{iB} = 2E_i^2(1 - \cos\theta_{iB})$$

squared transv. mom. wrt beam

k_t alg. at hadron colliders

2nd attempt: make it longitudinally boost-invariant

- Formulate in terms of rapidity (y), azimuth (ϕ), p_t

$$d_{ij} = \min(p_{ti}^2, p_{tj}^2)\Delta R_{ij}^2\,, \qquad \Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

NB: not η_i, E_{ti}

- Beam distance becomes

$$d_{iB} = p_{ti}^2$$

squared transv. mom. wrt beam

Catani, Dokshitzer, Seymour & Webber '93

Apart from measures, just like e^+e^- alg.
Known as **exclusive k_t algorithm**.

Problem: at hadron collider, no single fixed scale (as in Q in e^+e^-). So how do you choose d_{cut}?

See e.g. Seymour & Tevlin '06

k_t alg. at hadron colliders

3rd attempt: **inclusive k_t algorithm**

- Introduce angular radius R (NB: dimensionless!)

$$d_{ij} = \min(p_{ti}^2, p_{tj}^2)\frac{\Delta R_{ij}^2}{R^2}\,, \qquad d_{iB} = p_{ti}^2$$

- 1. Find smallest of d_{ij}, d_{iB}
  2. if ij, recombine them
  3. if iB, call i a jet and remove from list of particles
  4. repeat from step 1 until no particles left.

S.D. Ellis & Soper, '93; the simplest to use

Jets all separated by at least R on y, ϕ cylinder.

NB: number of jets not IR safe (soft jets near beam); number of jets above p_t cut **is** IR safe.

k_t is a form of Hierarchical Clustering

Fast Hierarchical Clustering and Other Applications of Dynamic Closest Pairs

David Eppstein
UC Irvine

We develop data structures for dynamic closest pair problems with arbitrary distance functions, that do not necessarily come from any geometric structure on the objects. Based on a technique previously used by the author for Euclidean closest pairs, we show how to insert and delete objects from an n-object set, maintaining the closest pair, in $O(n \log^2 n)$ time per update and $O(n)$ space. With quadratic space, we can instead use a quadtree-like structure to achieve an optimal time bound, $O(n)$ per update. We apply these data structures to hierarchical clustering, greedy matching, and TSP heuristics, and discuss other potential applications in machine learning, Gröbner bases, and local improvement algorithms for partition and placement problems. Experiments show our new methods to be faster in practice than previously used heuristics.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms**]: Nonnumeric Algorithms

General Terms: Closest Pair, Agglomerative Clustering

Additional Key Words and Phrases: TSP, matching, conga line data structure, quadtree, nearest neighbor heuristic

1. INTRODUCTION

Hierarchical clustering has long been a mainstay of statistical analysis, and clustering based methods have attracted attention in other fields: computational biology (reconstruction of evolutionary trees; tree-based multiple sequence alignment), scientific simulation (n-body problems), theoretical computer science (network design and nearest neighbor searching) and of course the web (hierarchical indices such as Yahoo). Many clustering methods have been devised and used in these applications, but less effort has gone into algorithmic speedups of these methods.

In this paper we identify and demonstrate speedups for a key subroutine used in several clustering algorithms, that of maintaining closest pairs in a dynamic set of objects. We also describe several other applications or potential applications of the

Idea behind k_t alg. is to be found over and over in many areas of (computer) science.

Sequential recombination

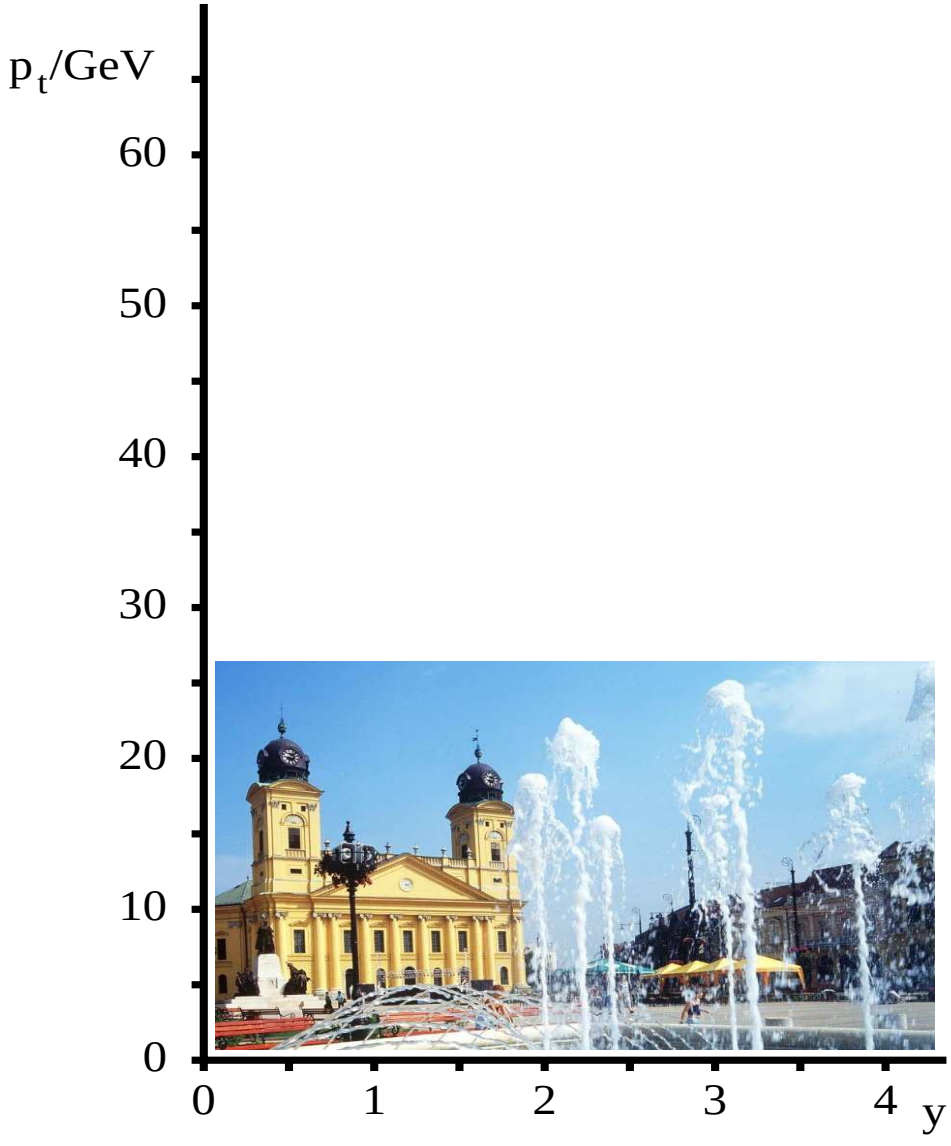
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



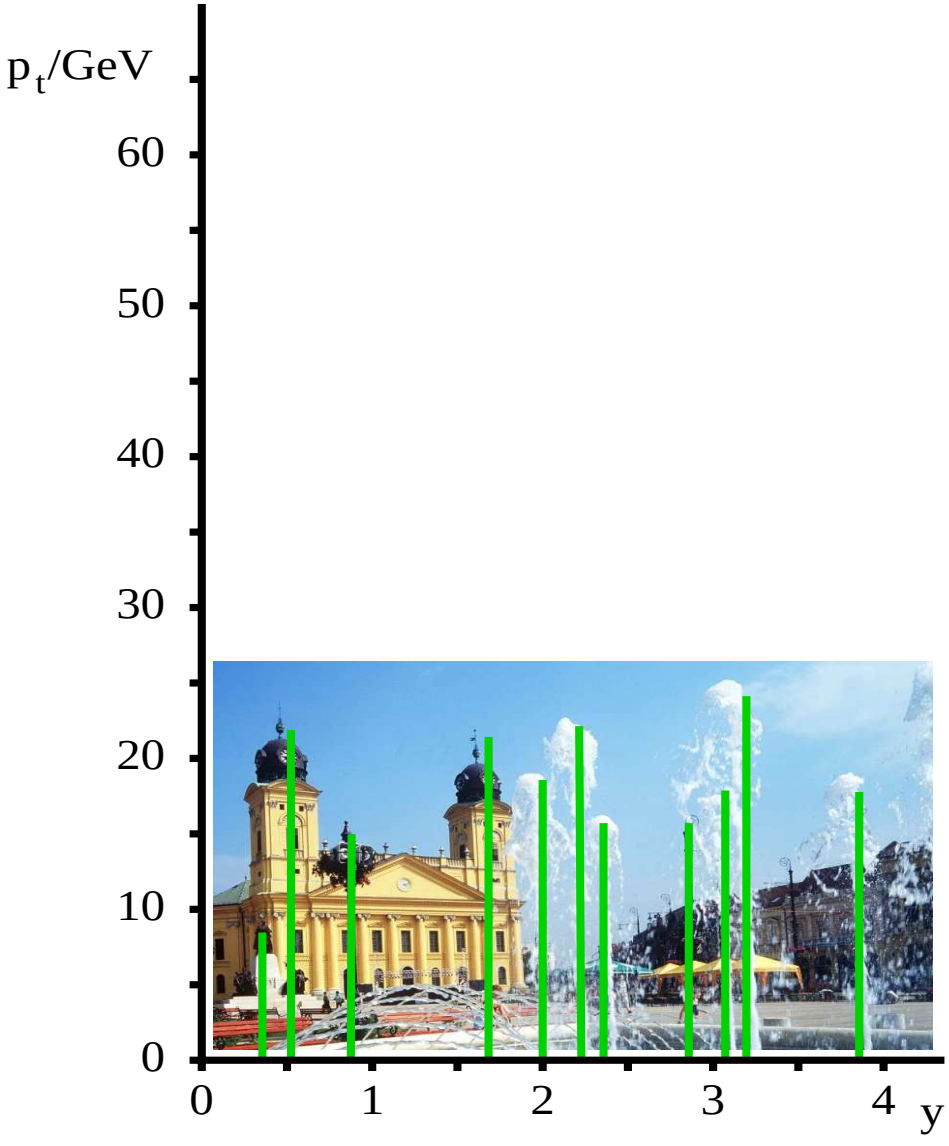
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



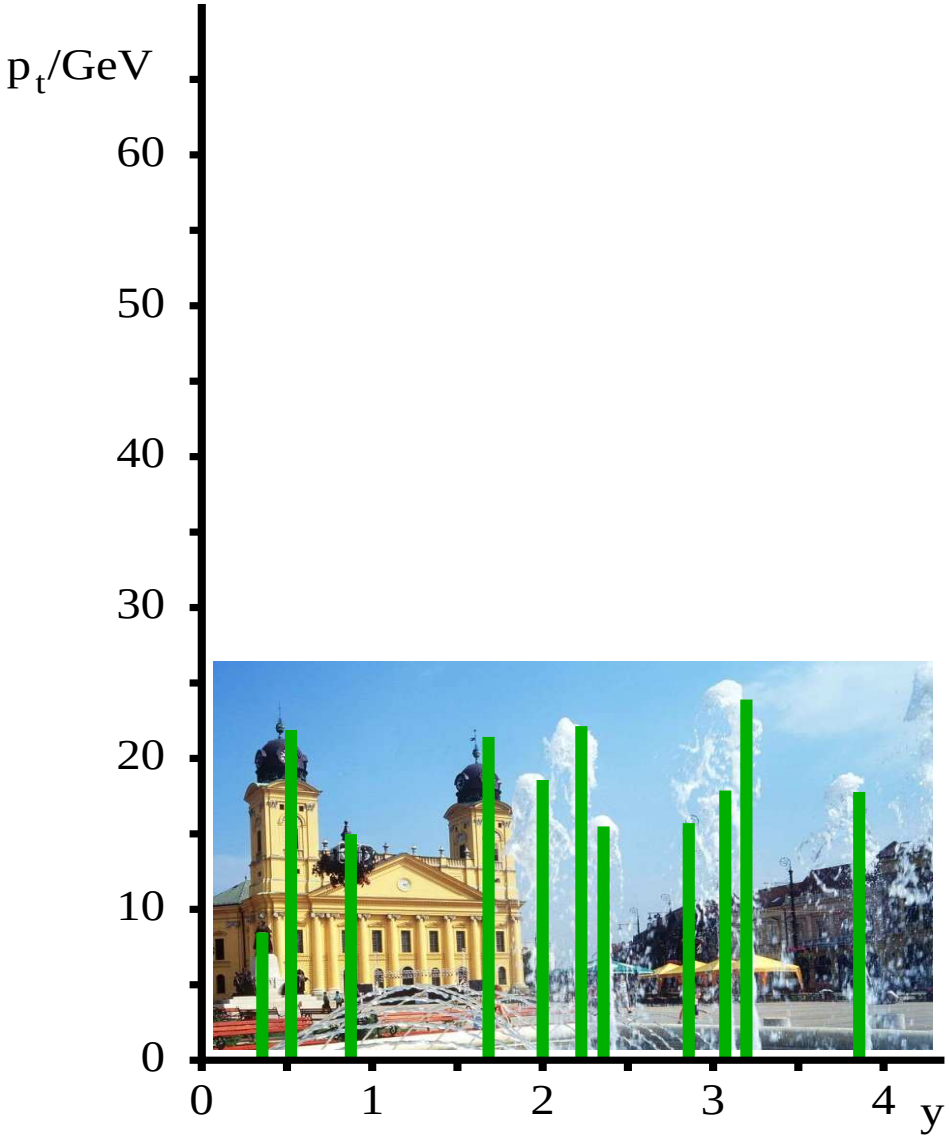
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



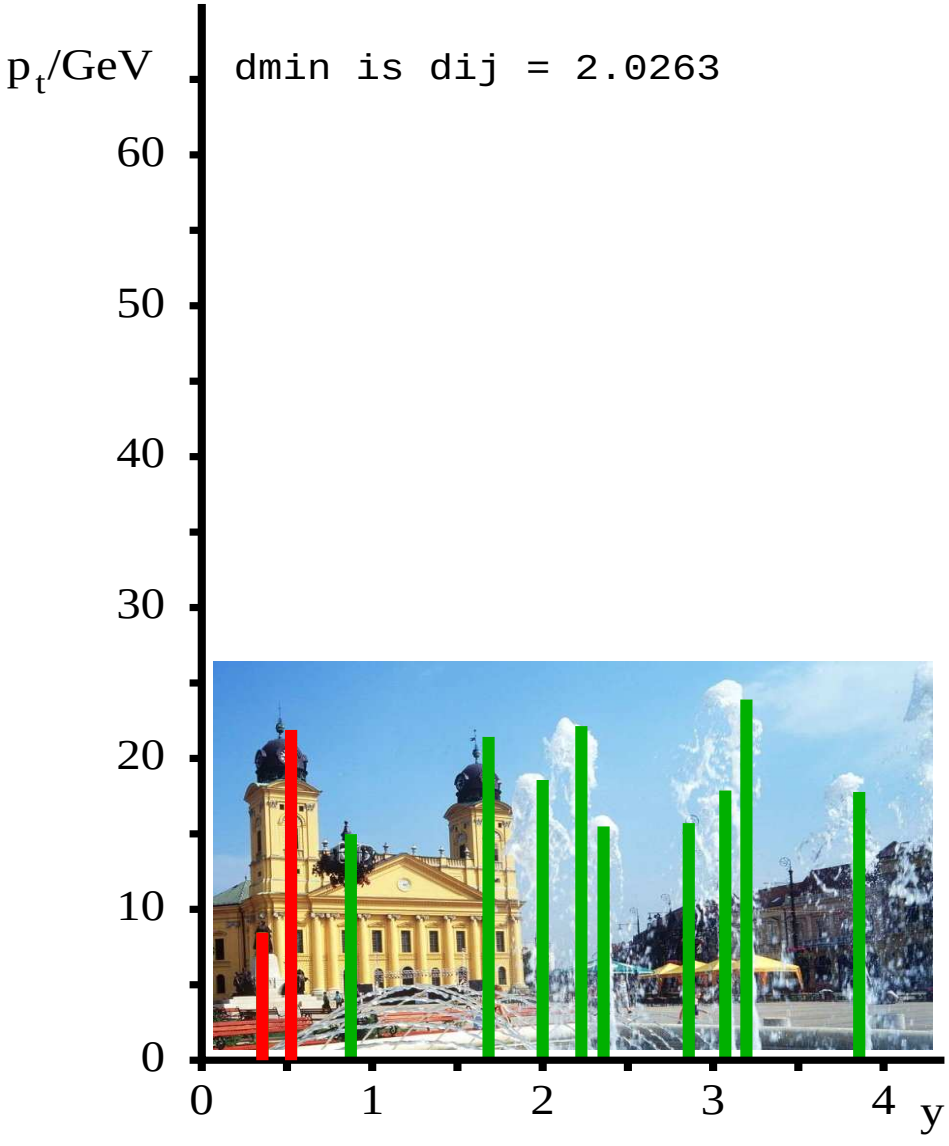
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



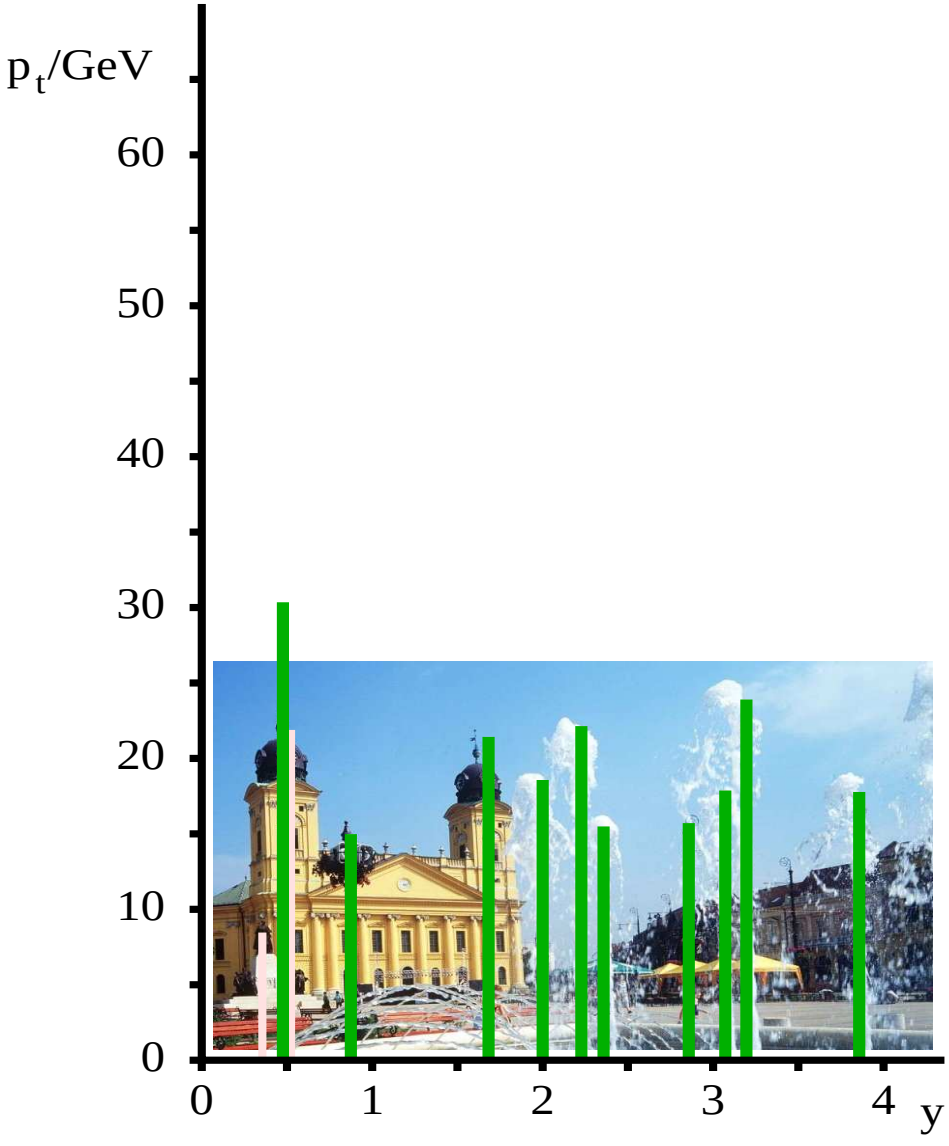
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



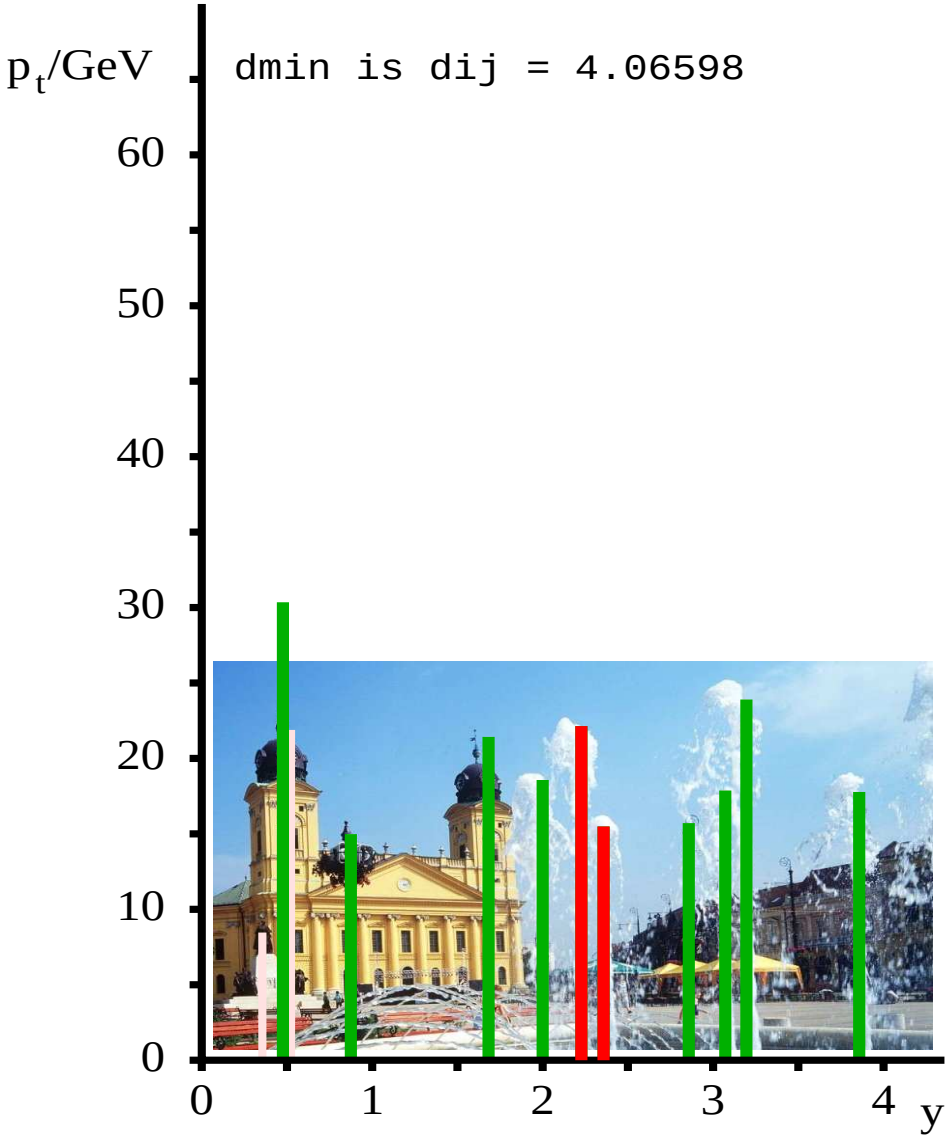
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



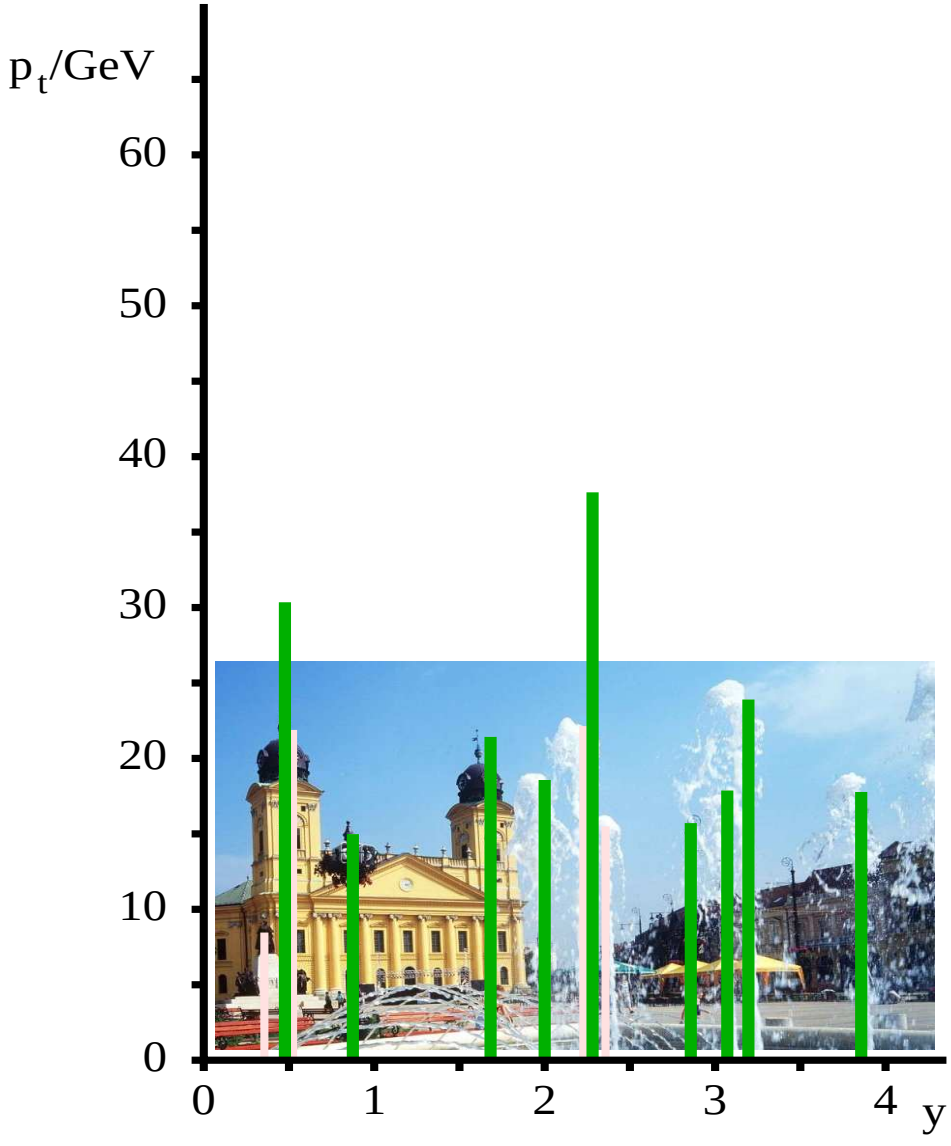
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



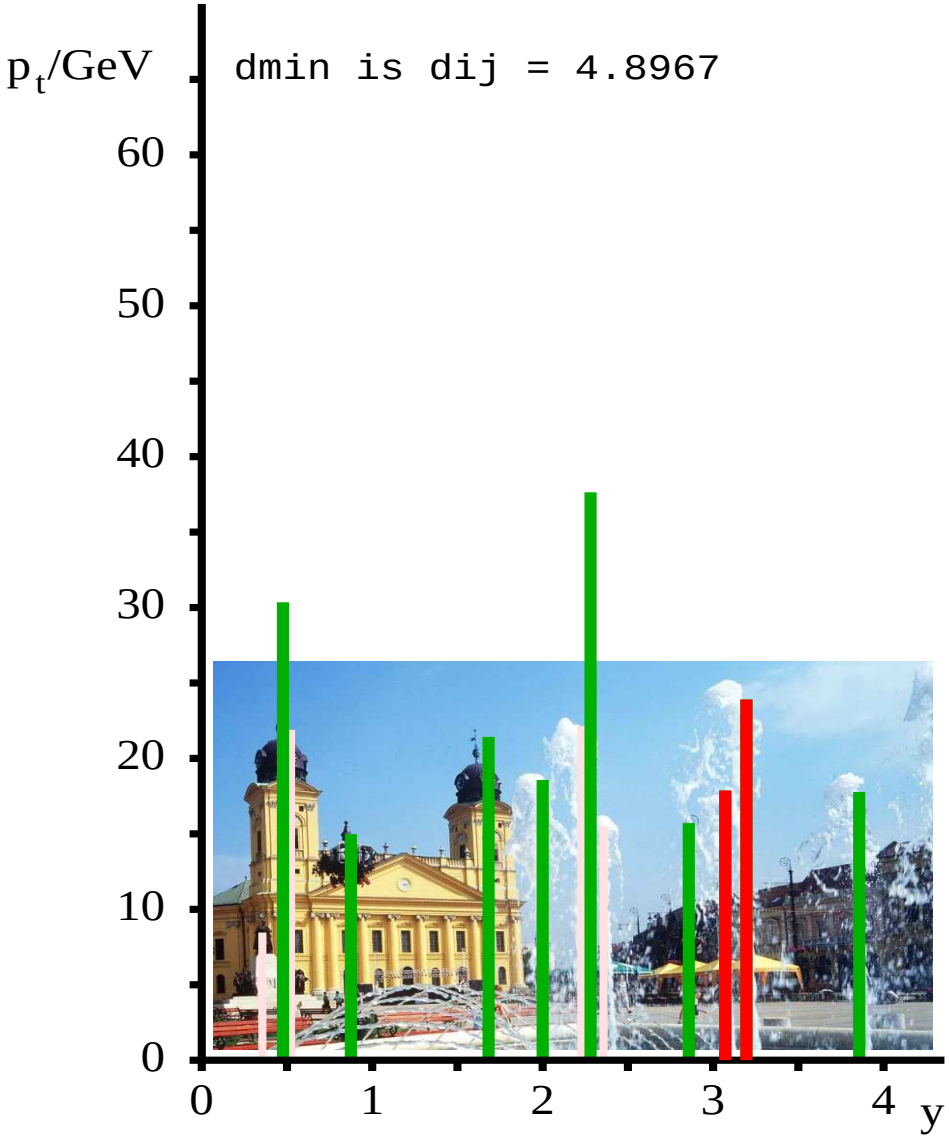
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



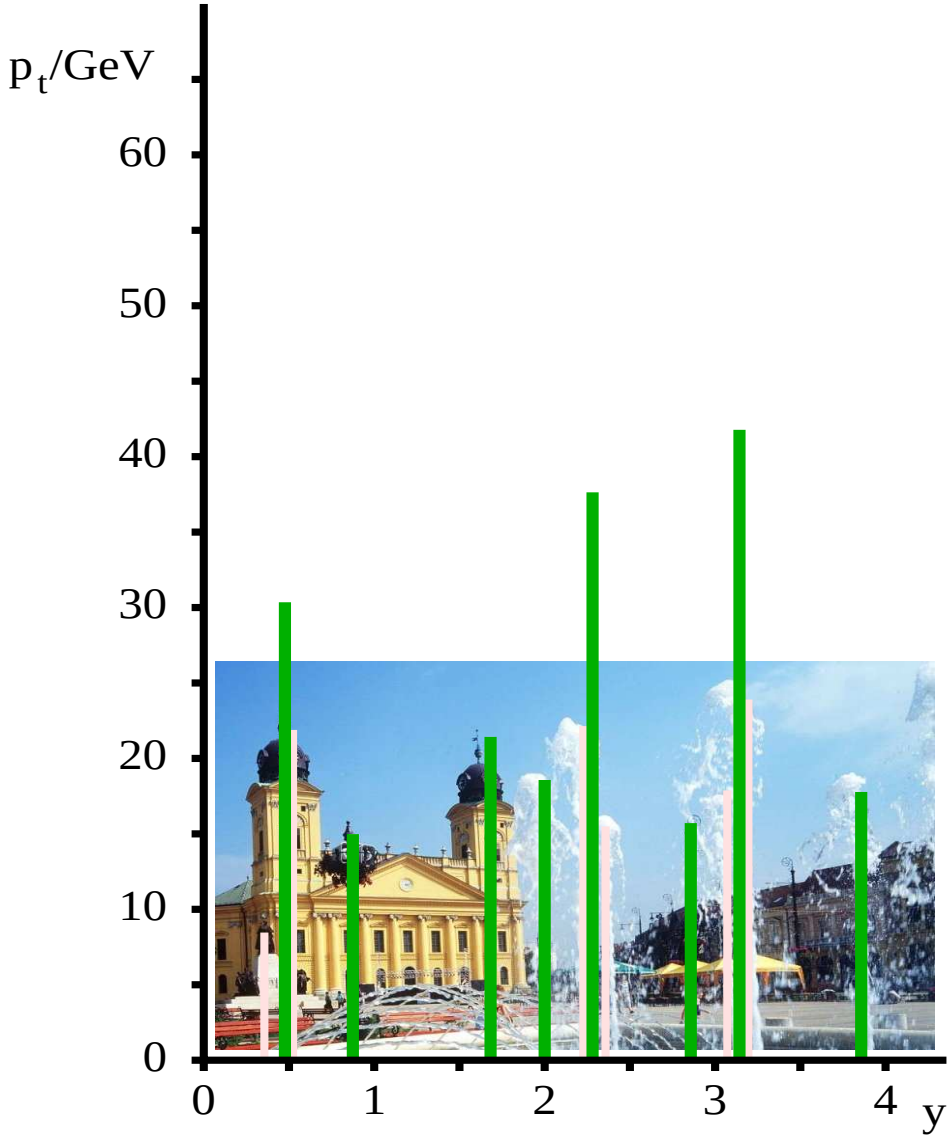
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



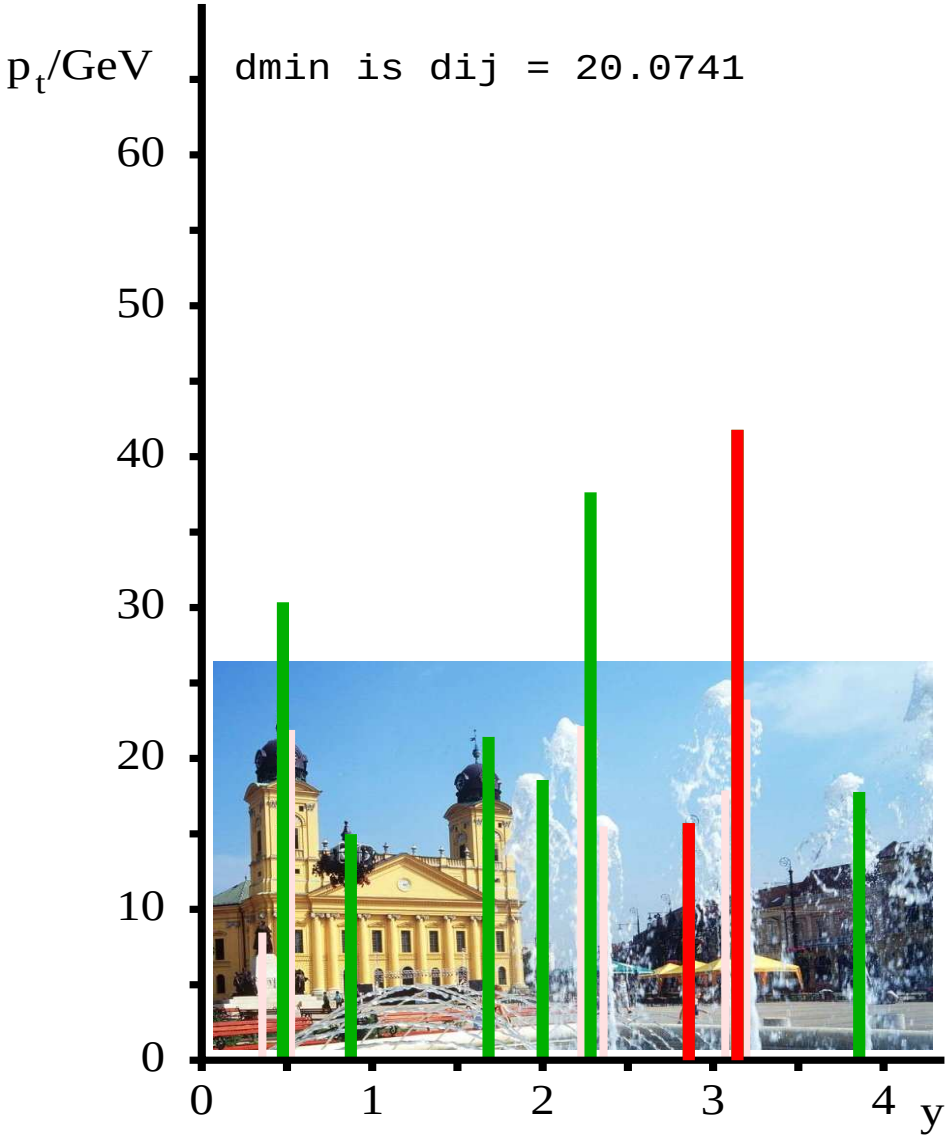
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



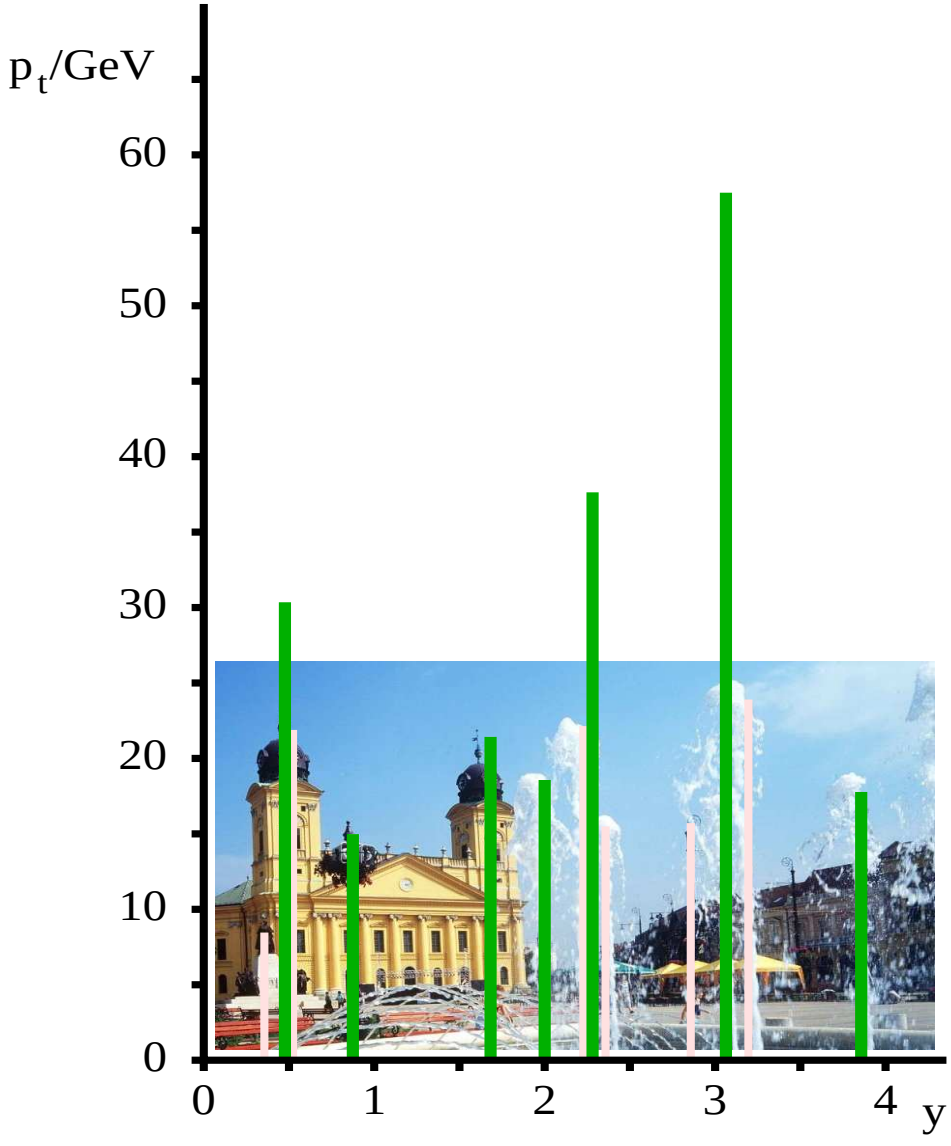
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



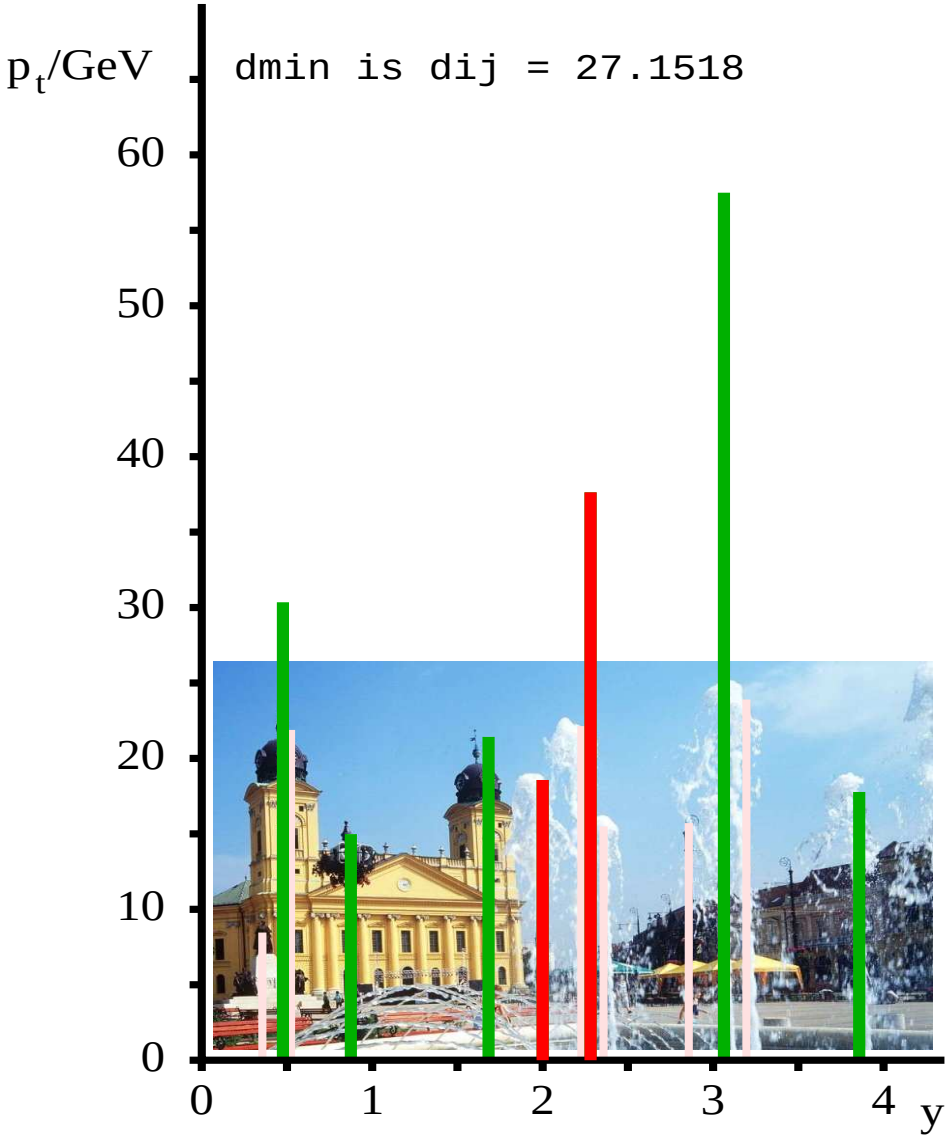
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



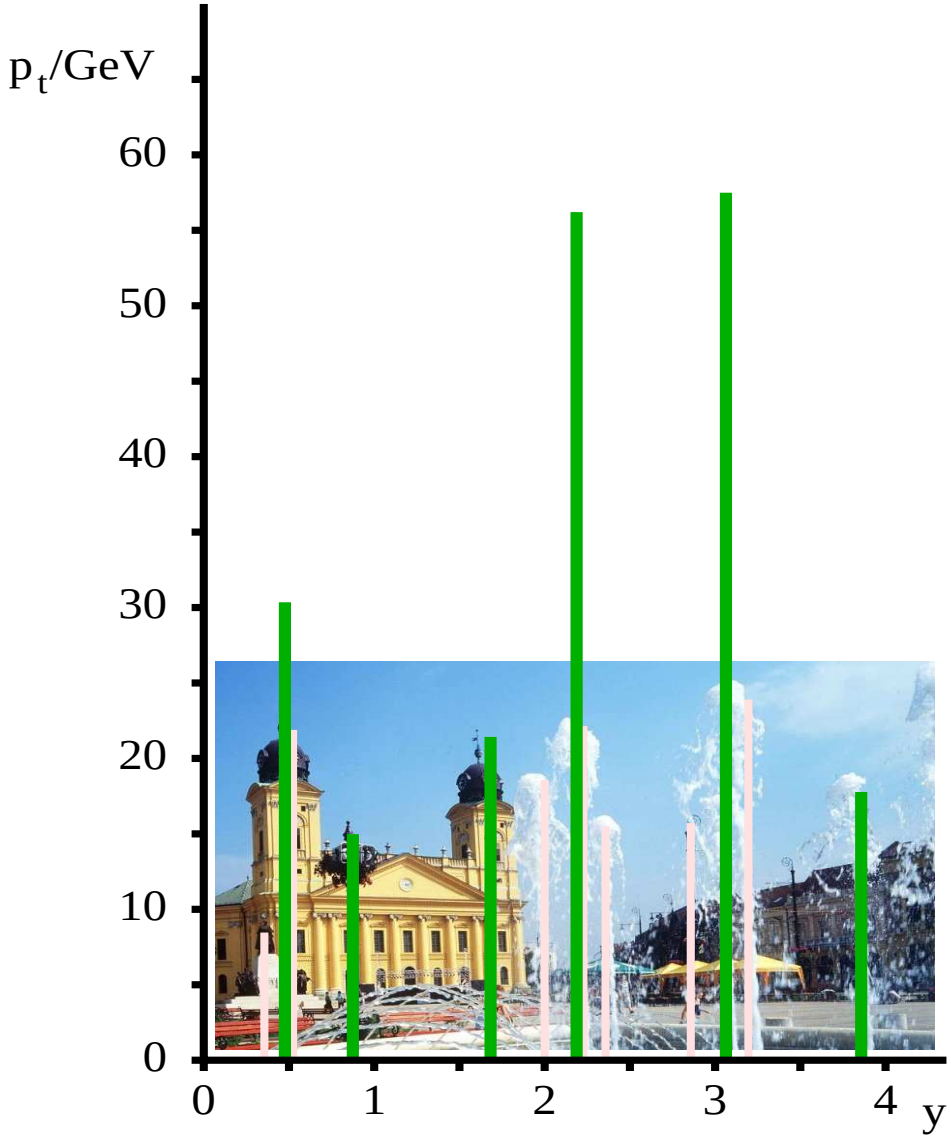
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



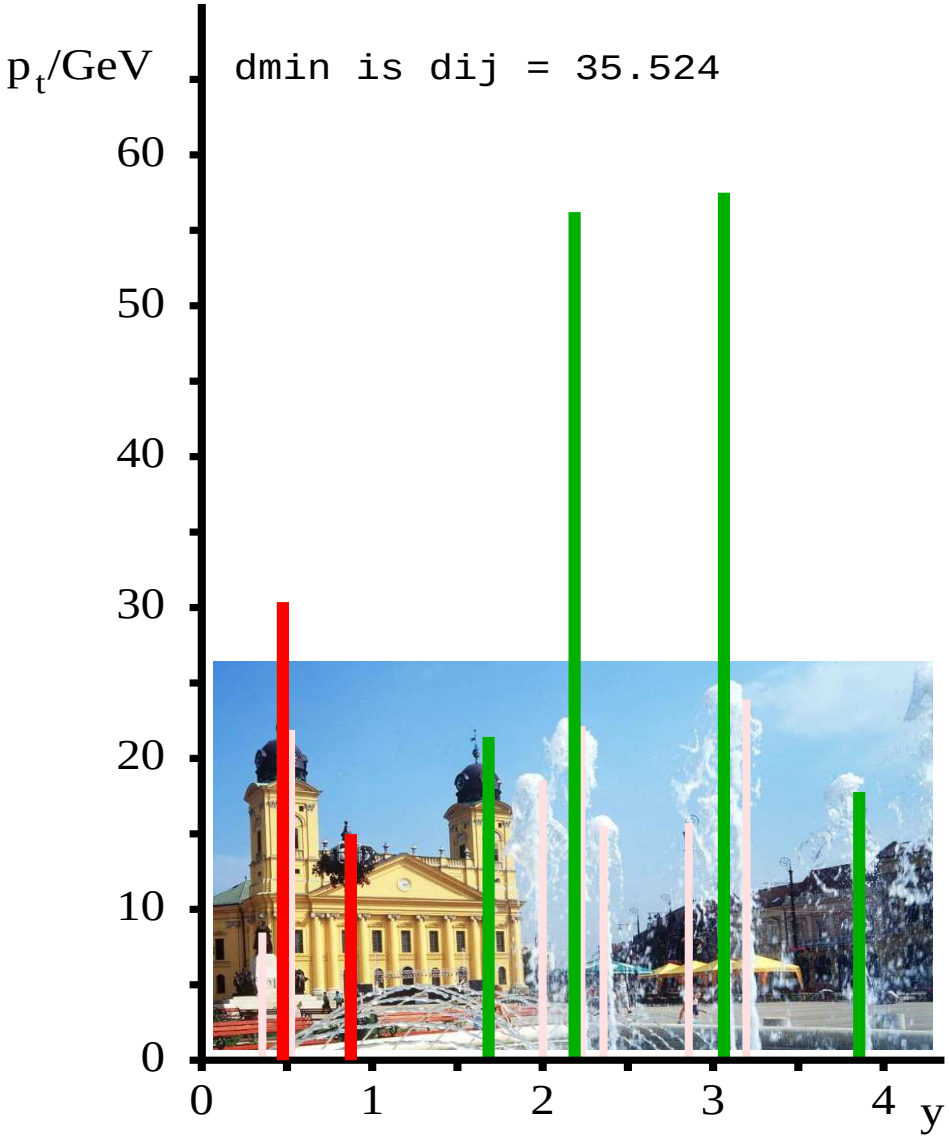
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



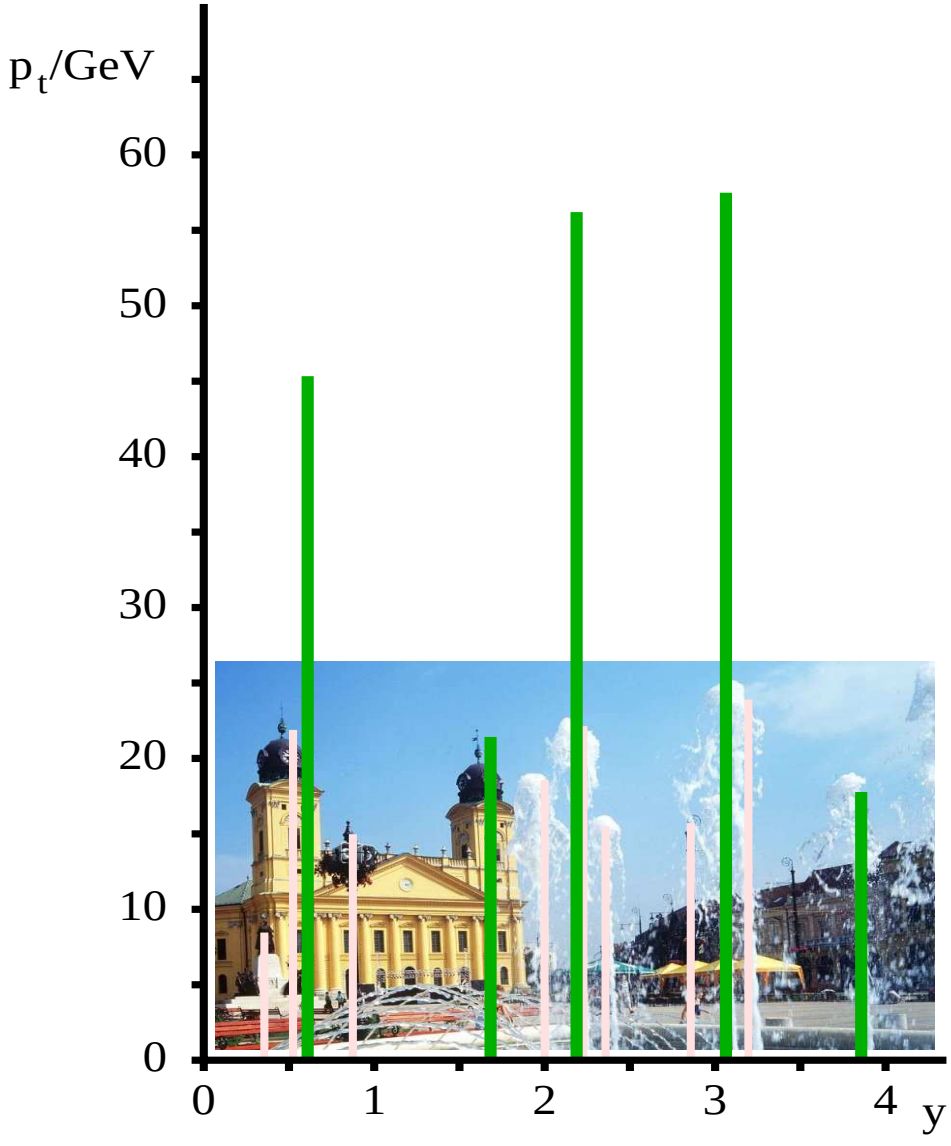
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



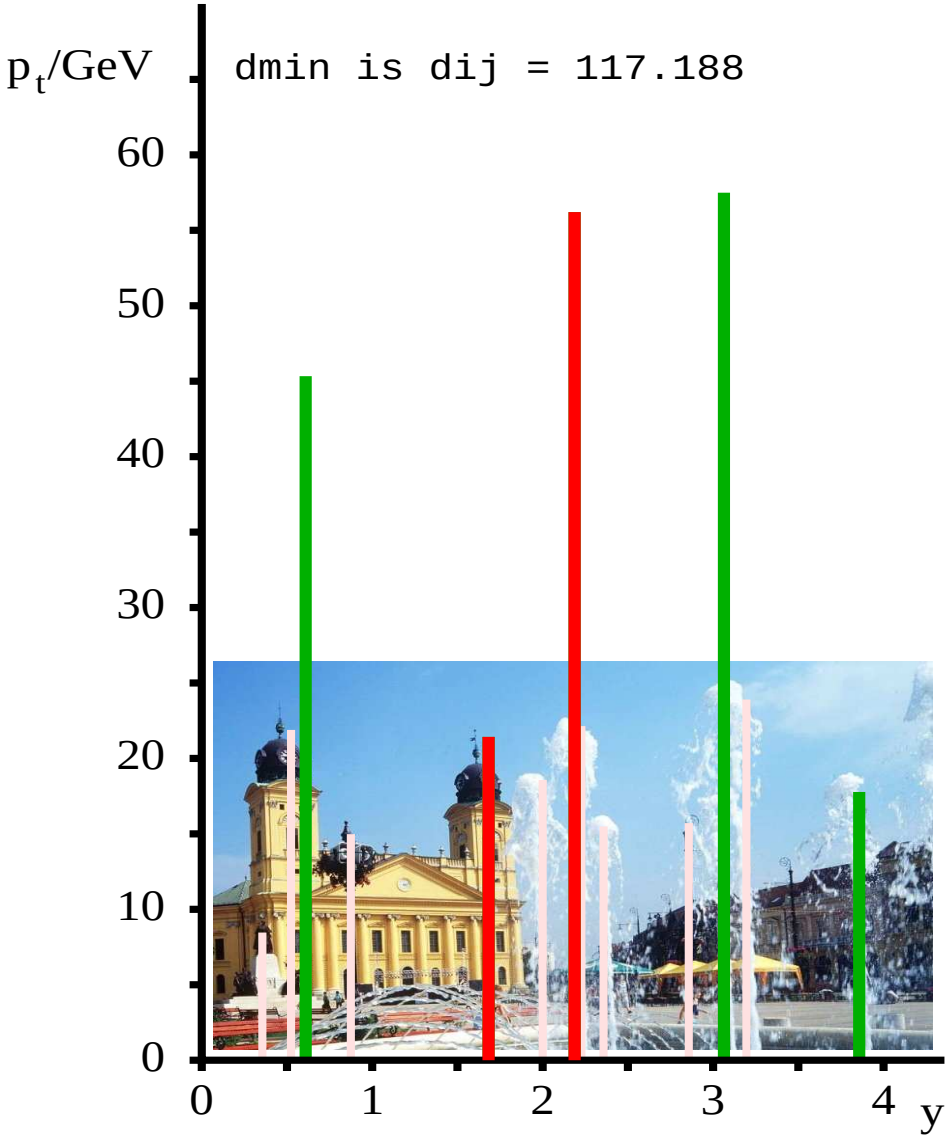
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



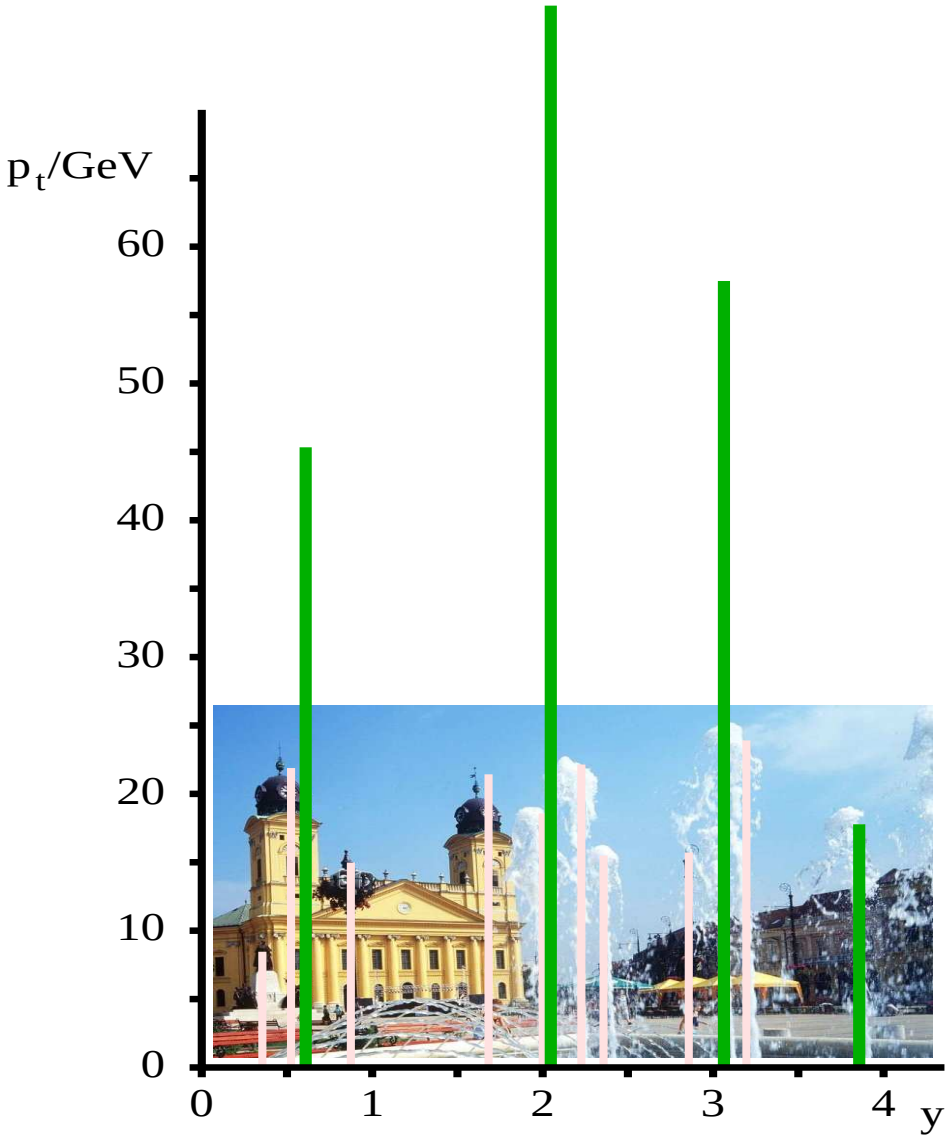
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



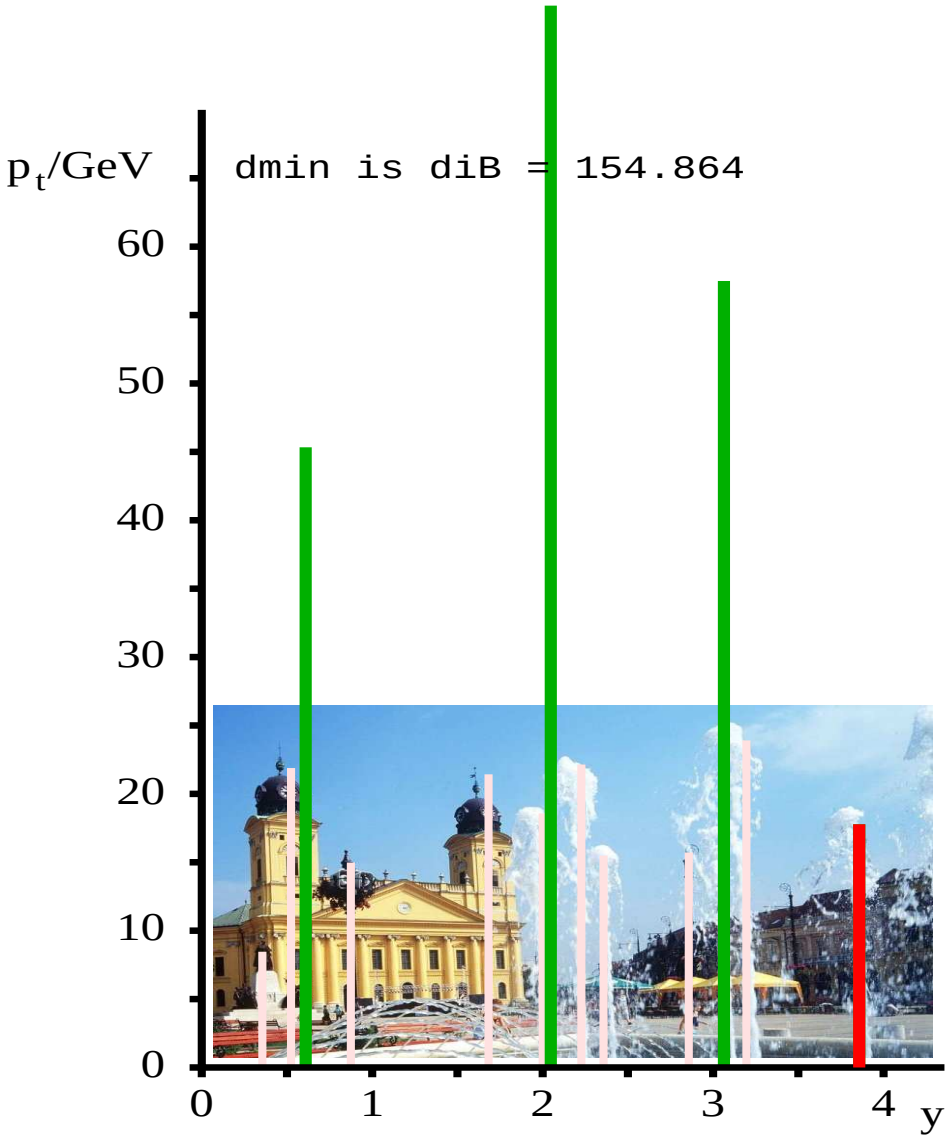
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



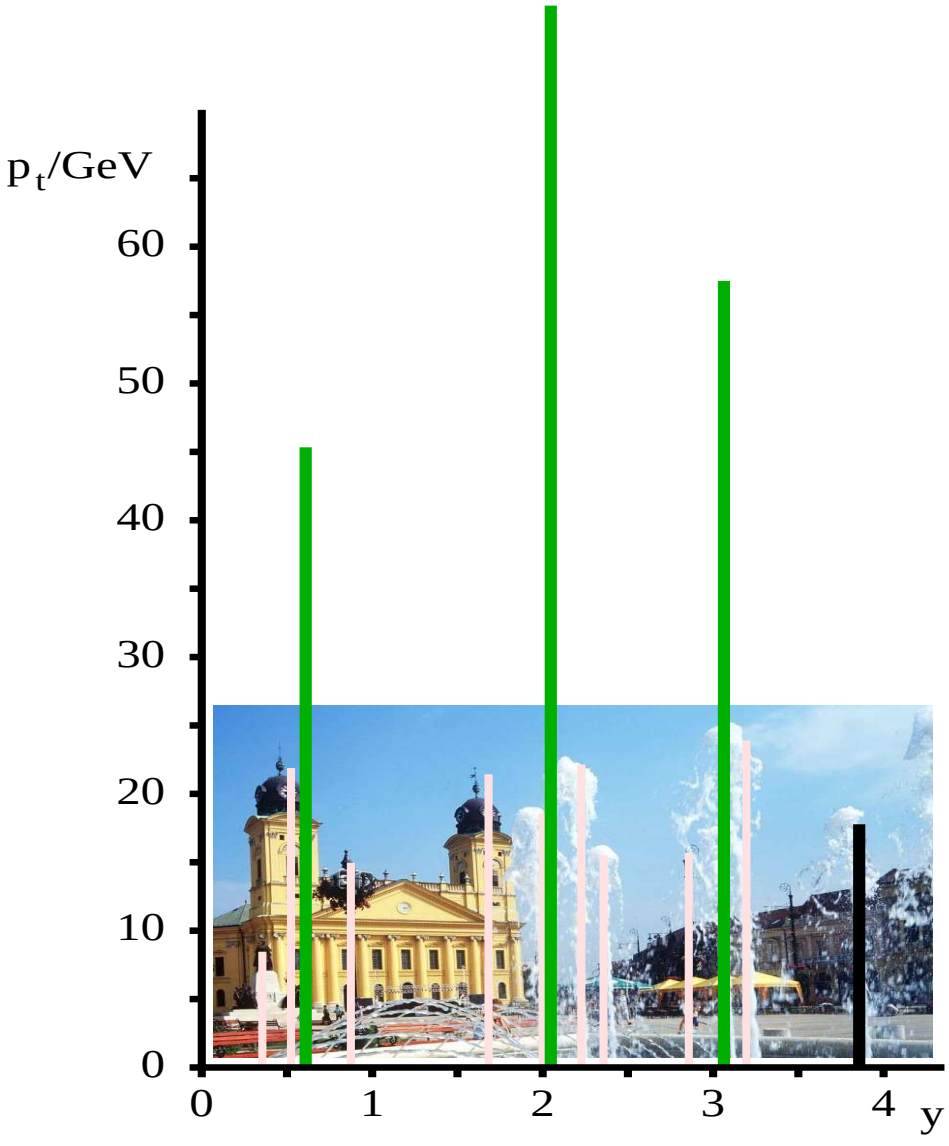
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



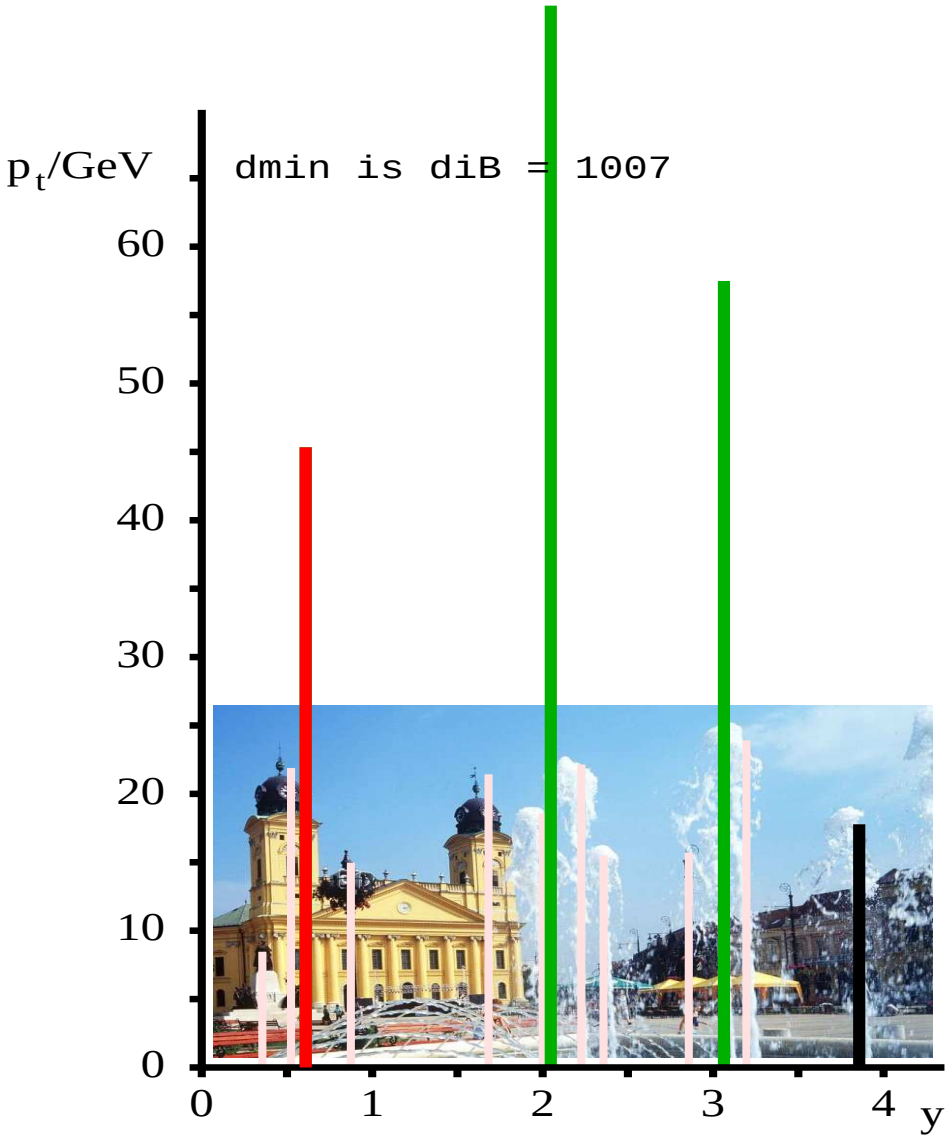
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



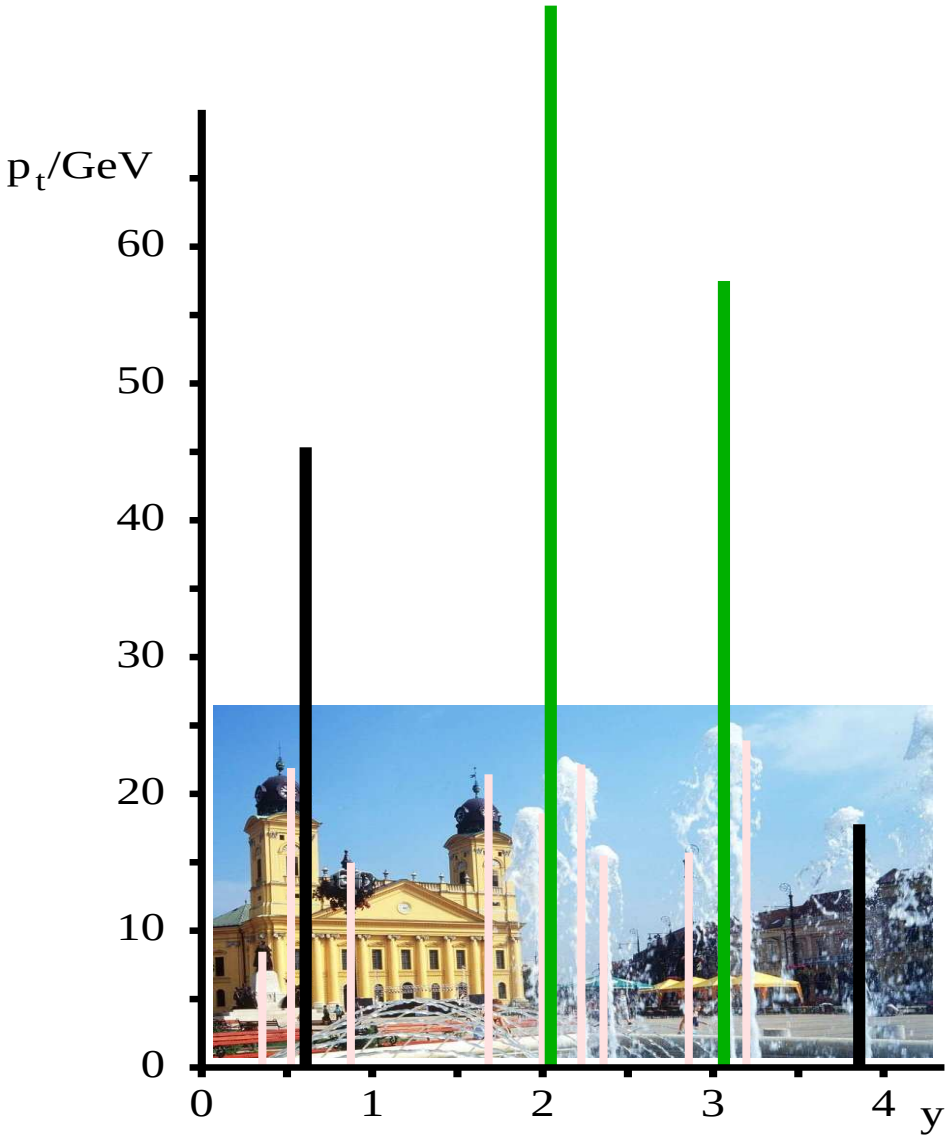
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



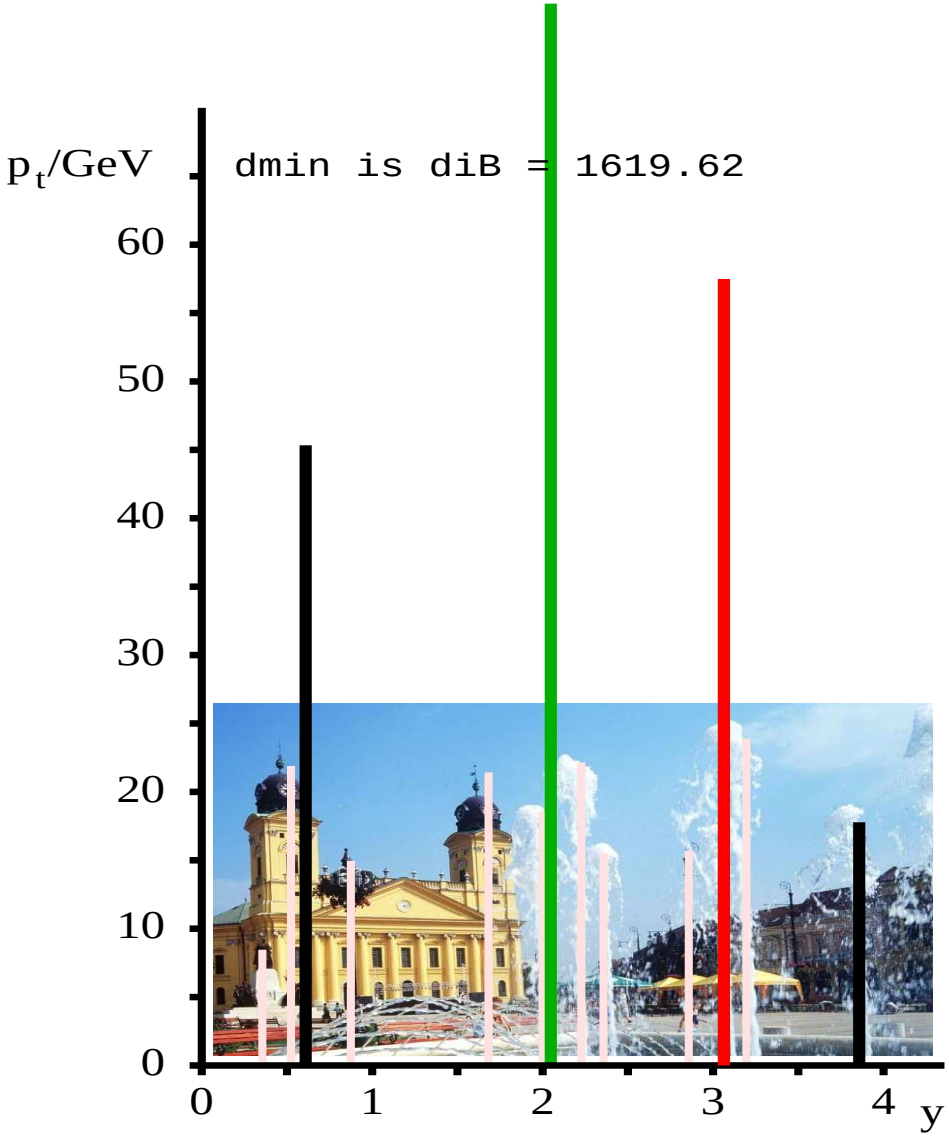
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



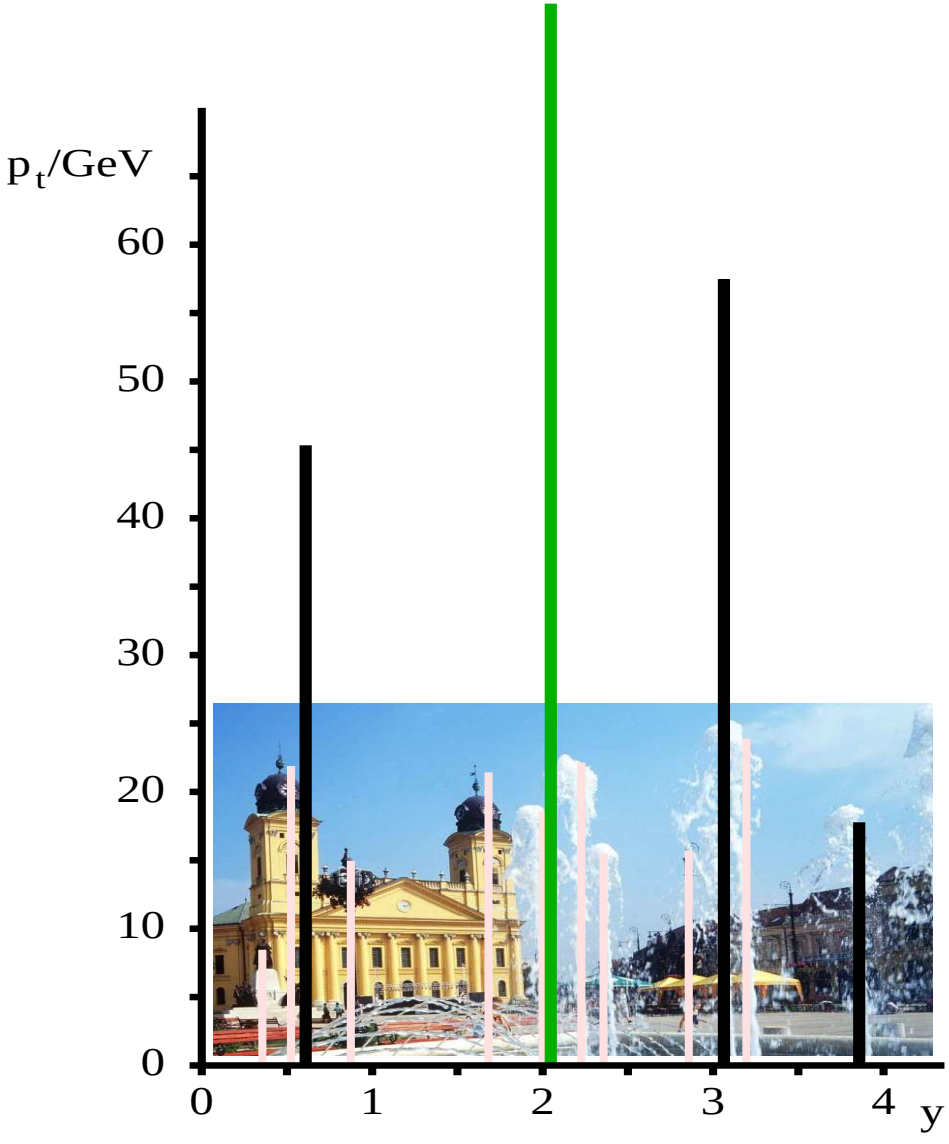
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



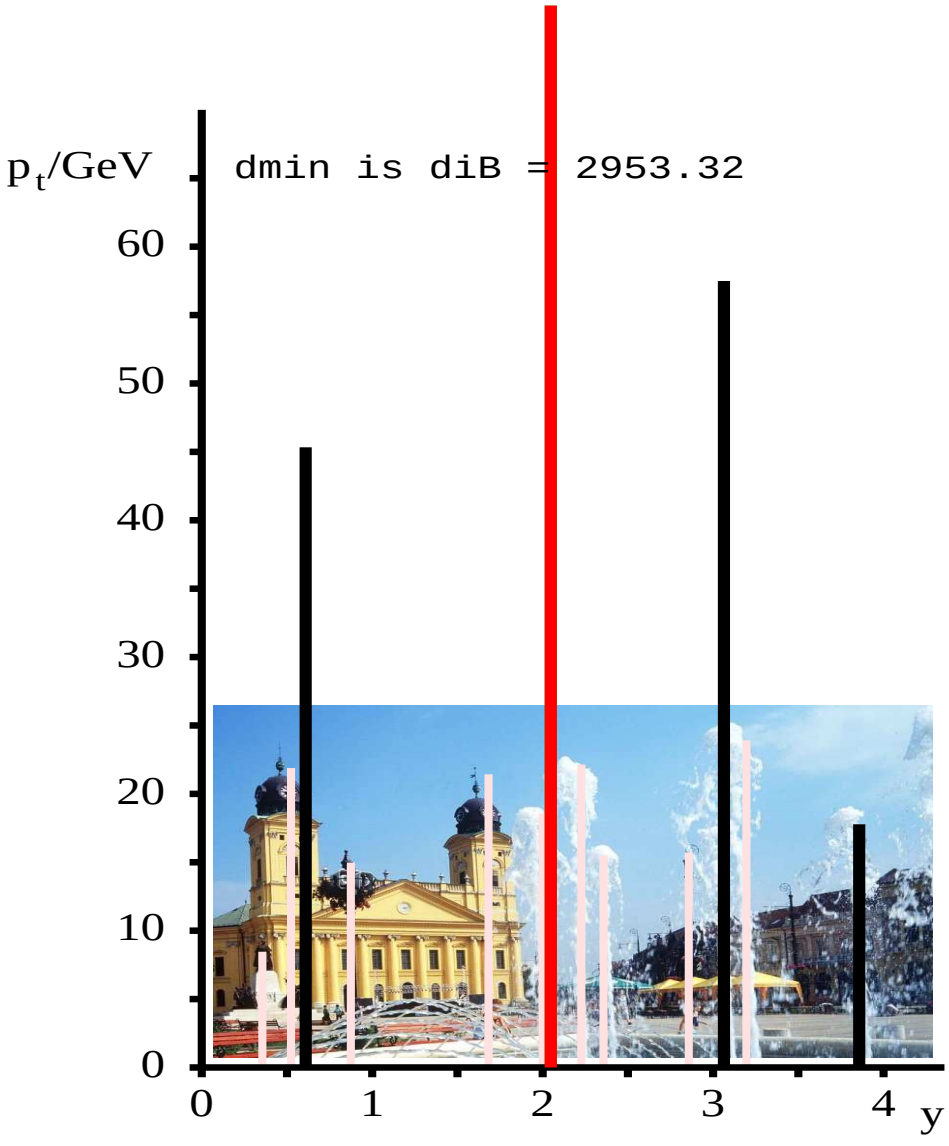
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



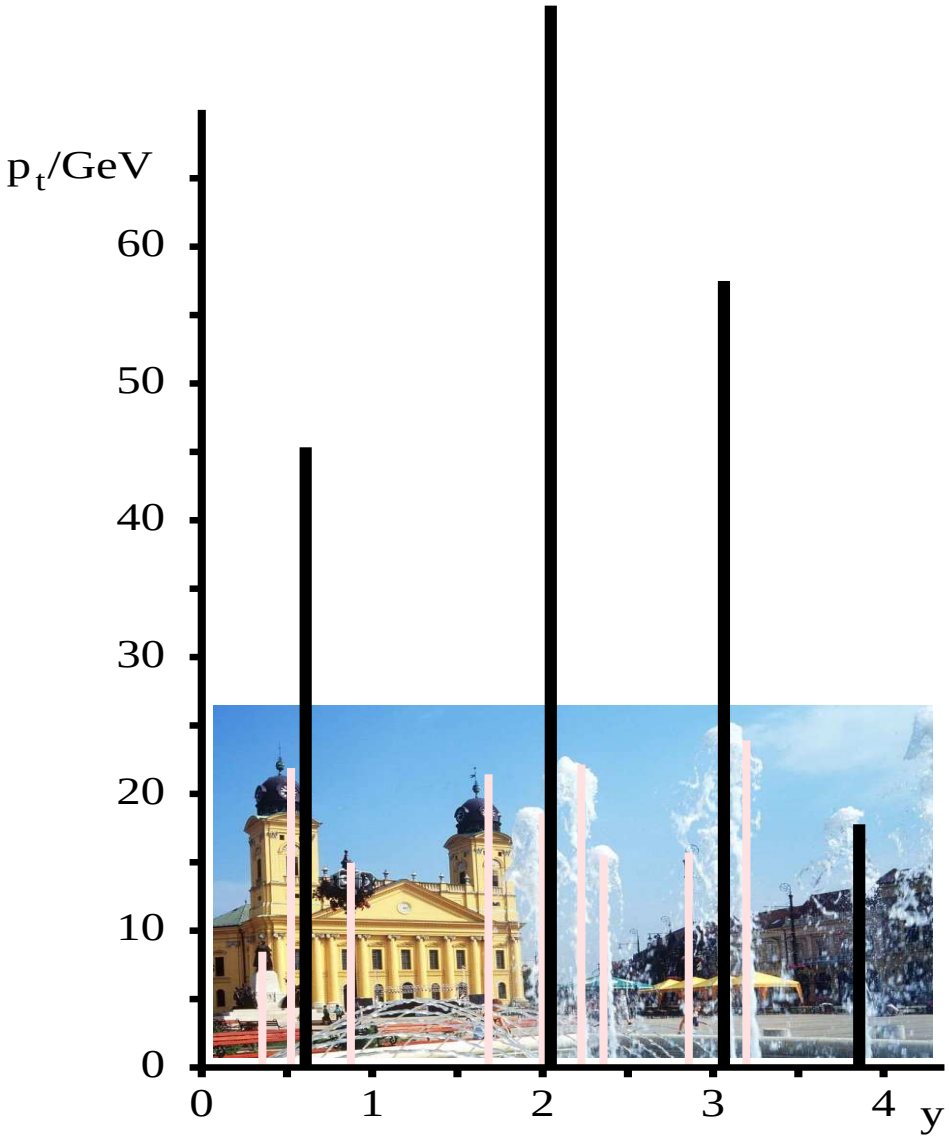
k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet
Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination variants

Cambridge/Aachen: *the simplest of hadron-collider algorithms*

- Recombine pair of objects closest in ΔR_{ij}
- Repeat until all $\Delta R_{ij} > R$ — remaining objects are jets

Dokshitzer, Leder, Moretti, Webber '97 (Cambridge): more involved e^+e^- form
Wobisch & Wengler '99 (Aachen): simple inclusive hadron-collider form

Anti-k_t: *formulated similarly to k_t, but with*

$$d_{ij} = \min\left(\frac{1}{k_{ti}^2}, \frac{1}{k_{tj}^2}\right)\frac{\Delta R_{ij}^2}{R^2}, \qquad d_{iB} = \frac{1}{k_{ti}^2}$$

Cacciari, GPS & Soyez, '08 [+ Delsart unpublished]
privileges clustering with *hard* particles first

Privileging different divergences $\Leftrightarrow$ different jets; more later...

Sequential recombination variants

Cambridge/Aachen: *the simplest of hadron-collider algorithms*

- Recombine pair of objects closest in ΔR_{ij}
- Repeat until all $\Delta R_{ij} > R$ — remaining objects are jets

Dokshitzer, Leder, Moretti, Webber '97 (Cambridge): more involved e^+e^- form
Wobisch & Wengler '99 (Aachen): simple inclusive hadron-collider form

Anti-k_t: *formulated similarly to k_t, but with*

$$d_{ij} = \min\left(\frac{1}{k_{ti}^2}, \frac{1}{k_{tj}^2}\right)\frac{\Delta R_{ij}^2}{R^2}, \qquad d_{iB} = \frac{1}{k_{ti}^2}$$

Cacciari, GPS & Soyez, '08 [+ Delsart unpublished]
privileges clustering with *hard* particles first

Privileging different divergences $\Leftrightarrow$ different jets; more later...

Sequential recombination variants

Cambridge/Aachen: *the simplest of hadron-collider algorithms*

- Recombine pair of objects closest in ΔR_{ij}
- Repeat until all $\Delta R_{ij} > R$ — remaining objects are jets

Dokshitzer, Leder, Moretti, Webber '97 (Cambridge): more involved e^+e^- form
Wobisch & Wengler '99 (Aachen): simple inclusive hadron-collider form

Anti-k_t: *formulated similarly to k_t, but with*

$$d_{ij} = \min\left(\frac{1}{k_{ti}^2}, \frac{1}{k_{tj}^2}\right)\frac{\Delta R_{ij}^2}{R^2}, \qquad d_{iB} = \frac{1}{k_{ti}^2}$$

Cacciari, GPS & Soyez, '08 [+ Delsart unpublished]
privileges clustering with *hard* particles first

Privileging different divergences $\Leftrightarrow$ different jets; more later...

Yet more variants

Plenty more variants too, mostly in e^+e^-, e.g.

- JADE: $d_{ij} = m_{ij}^2/Q^2$ — the original seq. rec. alg.
- Geneva $d_{ij} = 8E_iE_j(1-\cos\theta_{ij})/9(E_i+E_j)^2$
- ARCLUS: perform $3 \to 2$ recombination

In pp, also have modifications of angular measure

- QCD-metric angular distance: $\Delta R_{ij}^2 \to 2(\cosh\Delta y_{ij} - \cos\Delta\phi_{ij})$

And beyond just momentum

- Flavour-k_t algorithm (e^+e^- and pp)

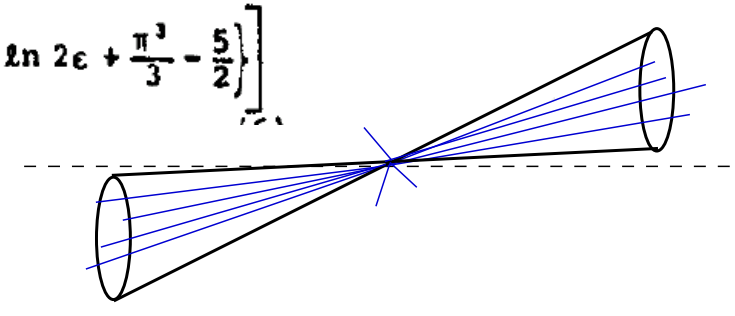
Cone algorithms

Cone Origins

First 'jet algorithm' dates back to Sterman and Weinberg (1977) — the original infrared-safe cross section:

To study jets, we consider the partial cross section $\sigma(E,\theta,\Omega,\epsilon,\delta)$ for e^+e^- hadron production events, in which all but a fraction $\epsilon \ll 1$ of the total e^+e^- energy E is emitted within some pair of oppositely directed cones of half-angle $\delta \ll 1$, lying within two fixed cones of solid angle Ω (with $\pi\delta^2 \ll \Omega \ll 1$) at an angle θ to the e^+e^- beam line. We expect this to be measur-

$$\sigma(E,\theta,\Omega,\epsilon,\delta) = (d\sigma/d\Omega)_0 \Omega\left[1 - (g_E^2/3\pi^2)\left\{3\ln\delta + 4\ln\delta\ln 2\epsilon + \frac{\pi^3}{3} - \frac{5}{2}\right\}\right]$$



Groundbreaking; good for 2 jets in e^+e^-; but never widely generalised

Cone algorithms today

Unifying idea: momentum flow within a cone only marginally modified by QCD branching

But cones come in many variants

Finding cones \ Processing	Progressive Removal	Split–Merge	Split–Drop
Seeded, Fixed (FC)	GetJet CellJet		
Seeded, Iterative (IC)	CMS Cone	JetClu (CDF)[†] ATLAS cone	
Seeded, It. + Midpoints (IC_{mp})		CDF MidPoint D0 Run II cone	PxCone
Seedless (SC)		SISCone	

[†] JetClu also has "ratcheting"

Cone algorithms today

Unifying idea: momentum flow within a cone only marginally modified by QCD branching

But cones come in many variants

Finding cones \ Processing	Progressive Removal	Split–Merge	Split–Drop
Seeded, Fixed (FC)	GetJet CellJet		
Seeded, Iterative (IC)	CMS Cone	JetClu (CDF)† ATLAS cone	
Seeded, It. + Midpoints (IC_{mp})		CDF MidPoint D0 Run II cone	PxCone
Seedless (SC)		SISCone	

† JetClu also has "ratcheting"

Cone algorithms today

Unifying idea: momentum flow within a cone only marginally modified by QCD branching

But cones come in many variants

Finding cones \ Processing	Progressive Removal	Split–Merge	Split–Drop
Seeded, Fixed (FC)	GetJet CellJet		
Seeded, Iterative (IC)	CMS Cone	JetClu (CDF)† ATLAS cone	
Seeded, It. + Midpoints (IC_{mp})		CDF MidPoint D0 Run II cone	PxCone
Seedless (SC)		SISCone	

† JetClu also has "ratcheting"

Common features in discussion of cones

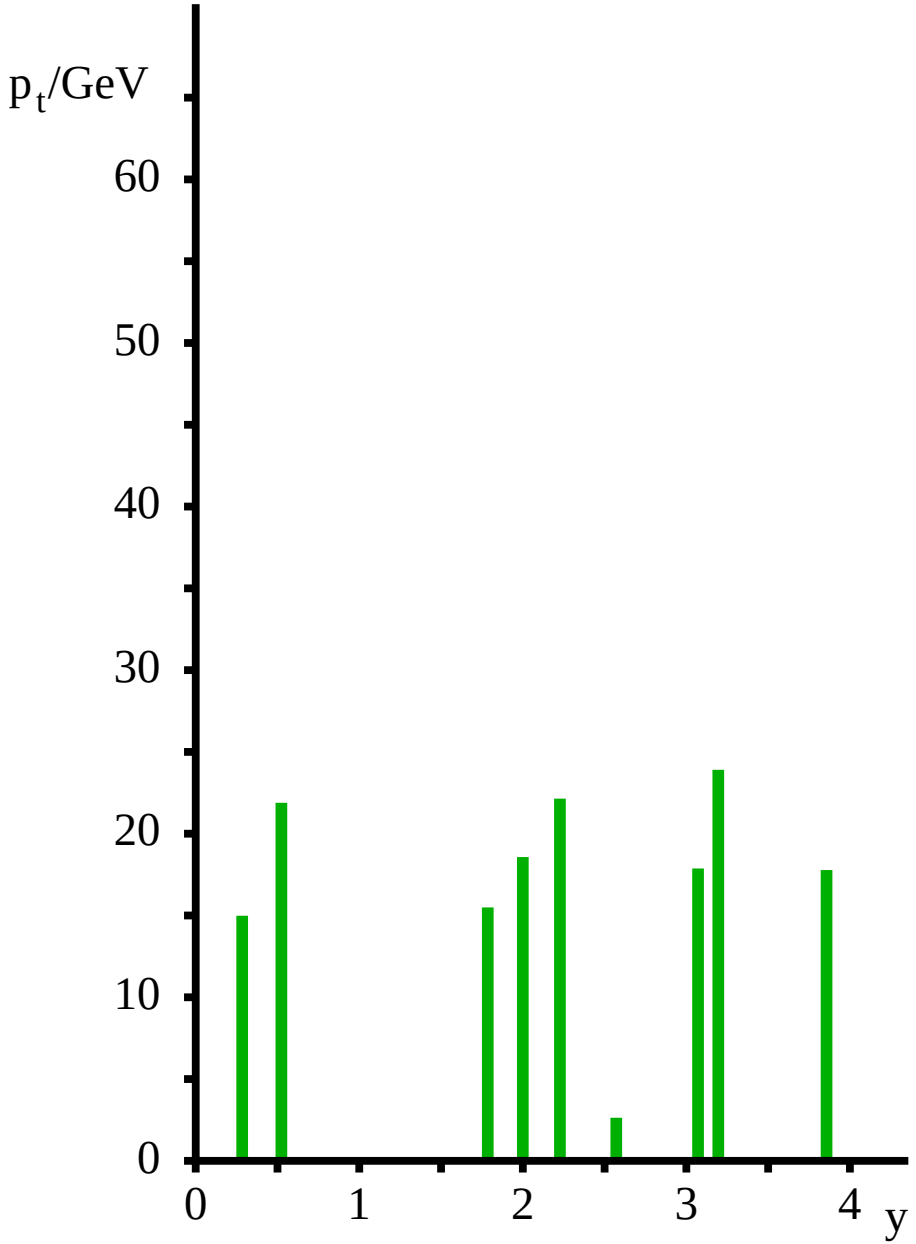
- Cones are always understood as circles in rapidity (y) and azimuth ϕ.
- A particle i is within the cone of radius R around the axis a if

$$\Delta R_{ia}^2 = (y_i - y_a)^2 + (\phi_i - \phi_a)^2 < R^2$$

The usual hadron collider variables

- We'll use $R = 0.7$ in the examples that follow
- And we'll use events all of whose particles are at $\phi = 0$, for simplicity

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

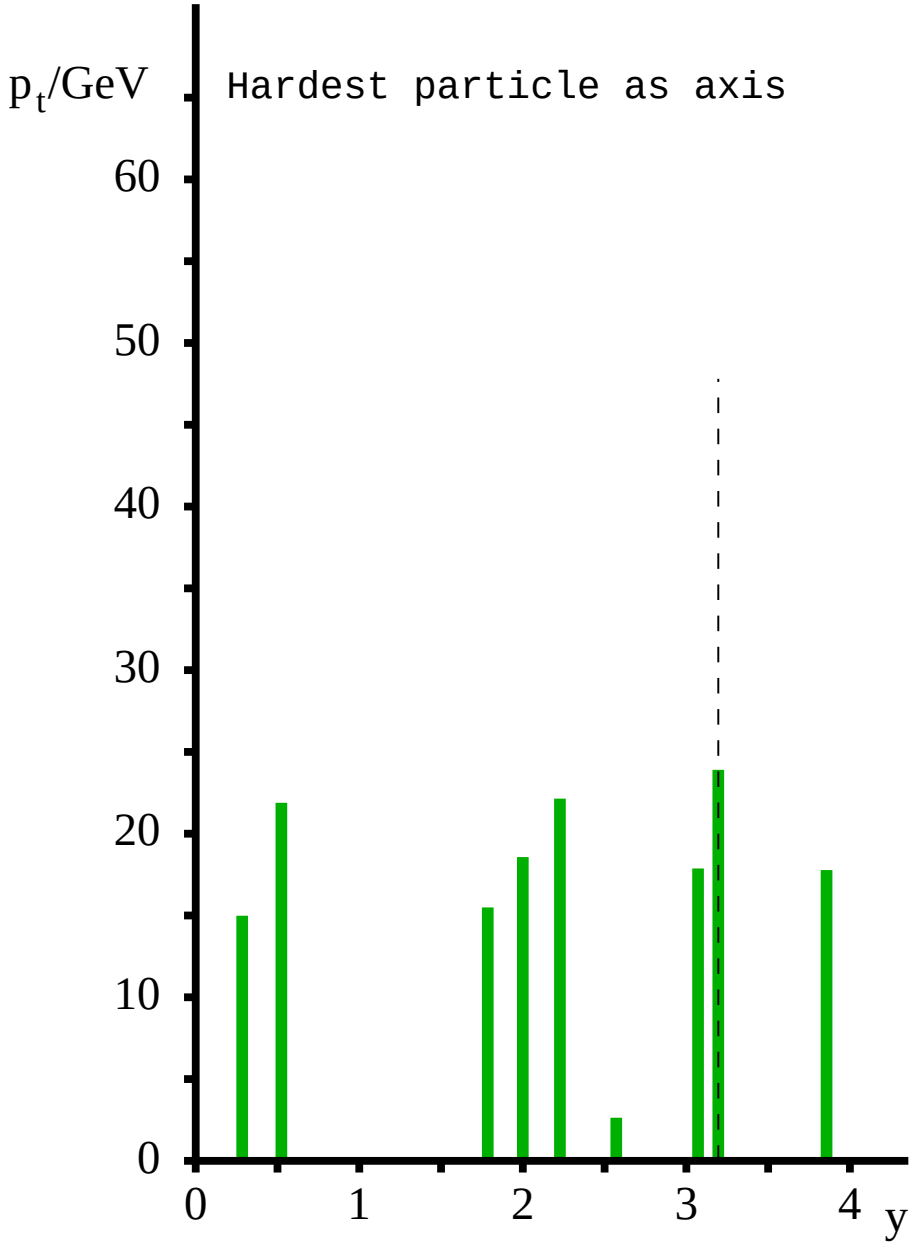
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe — more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

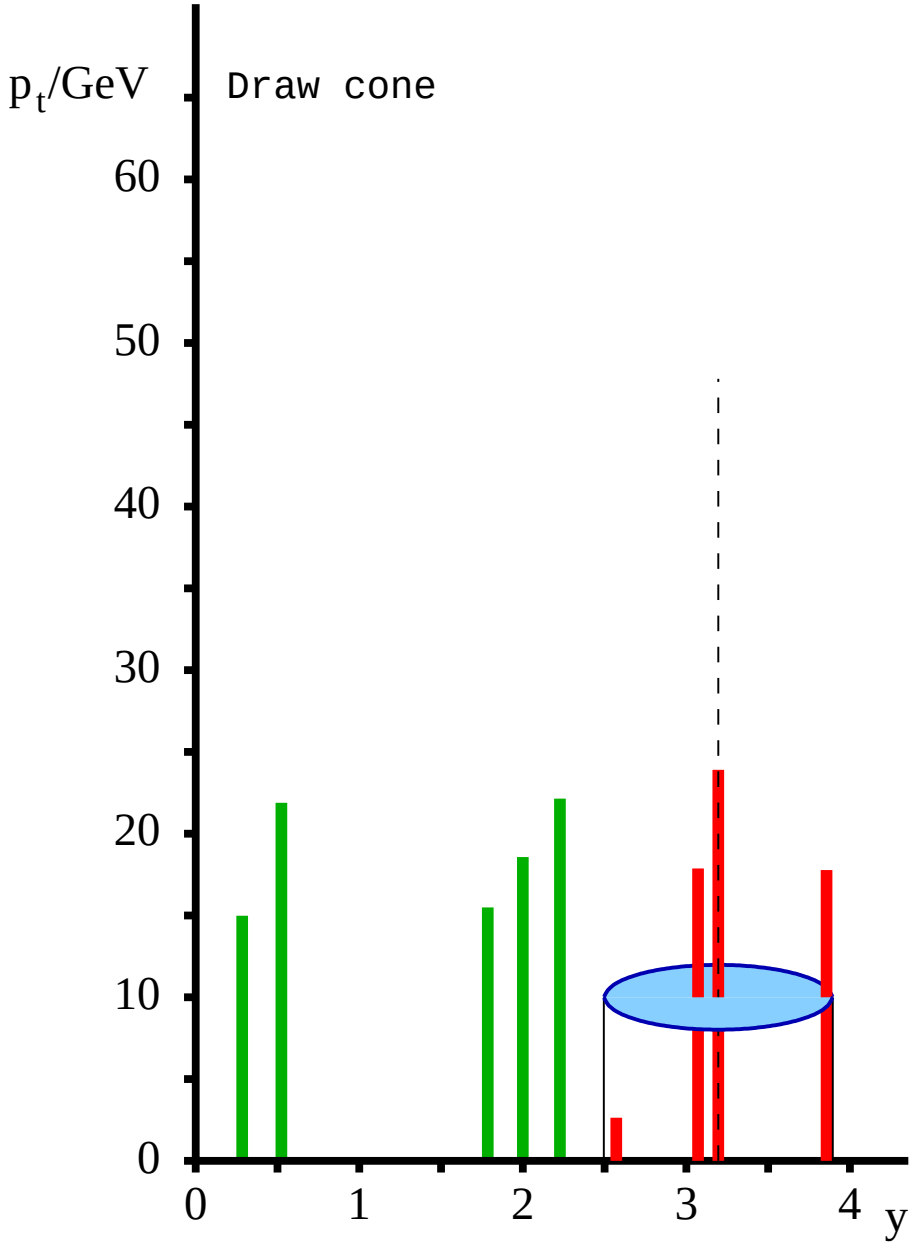
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ... more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

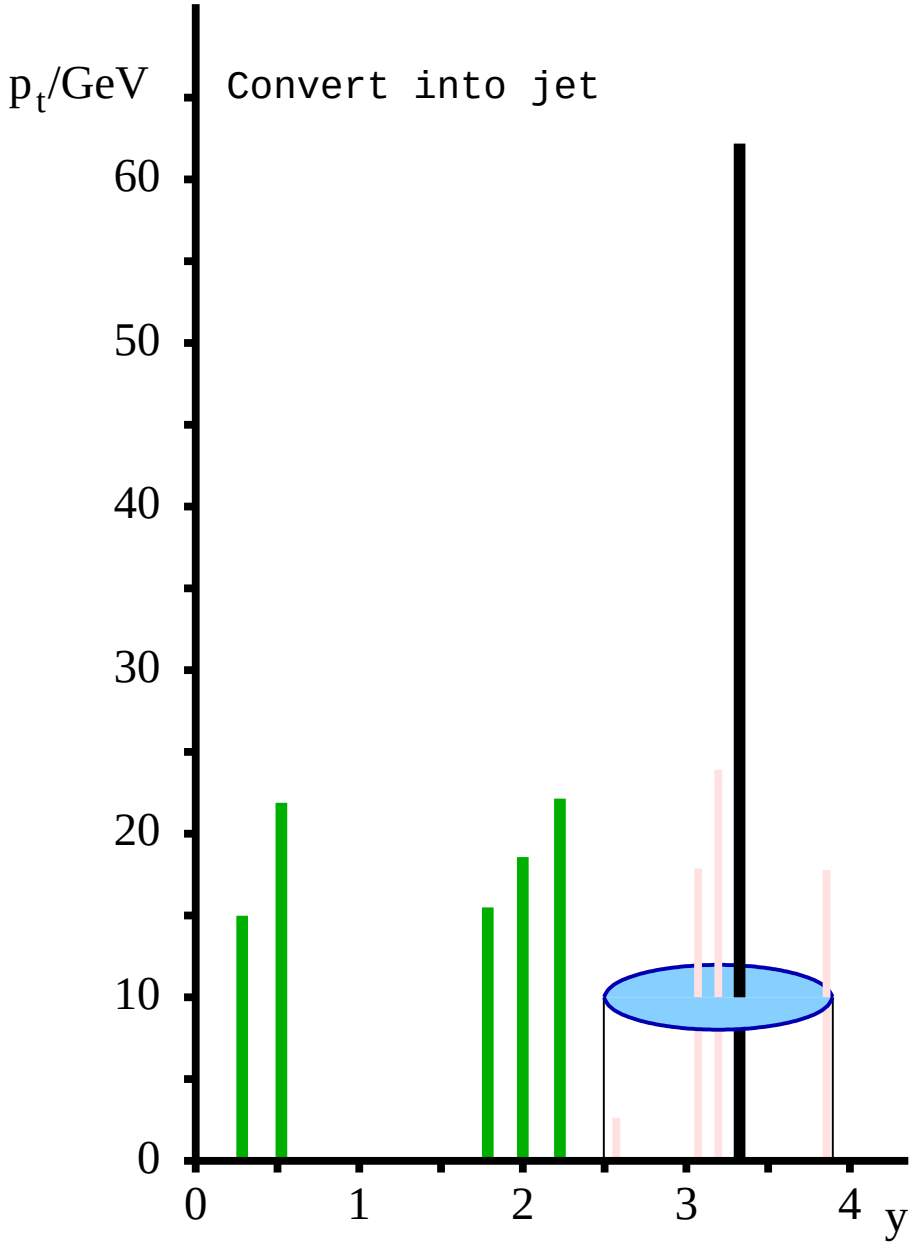
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ⇒ more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

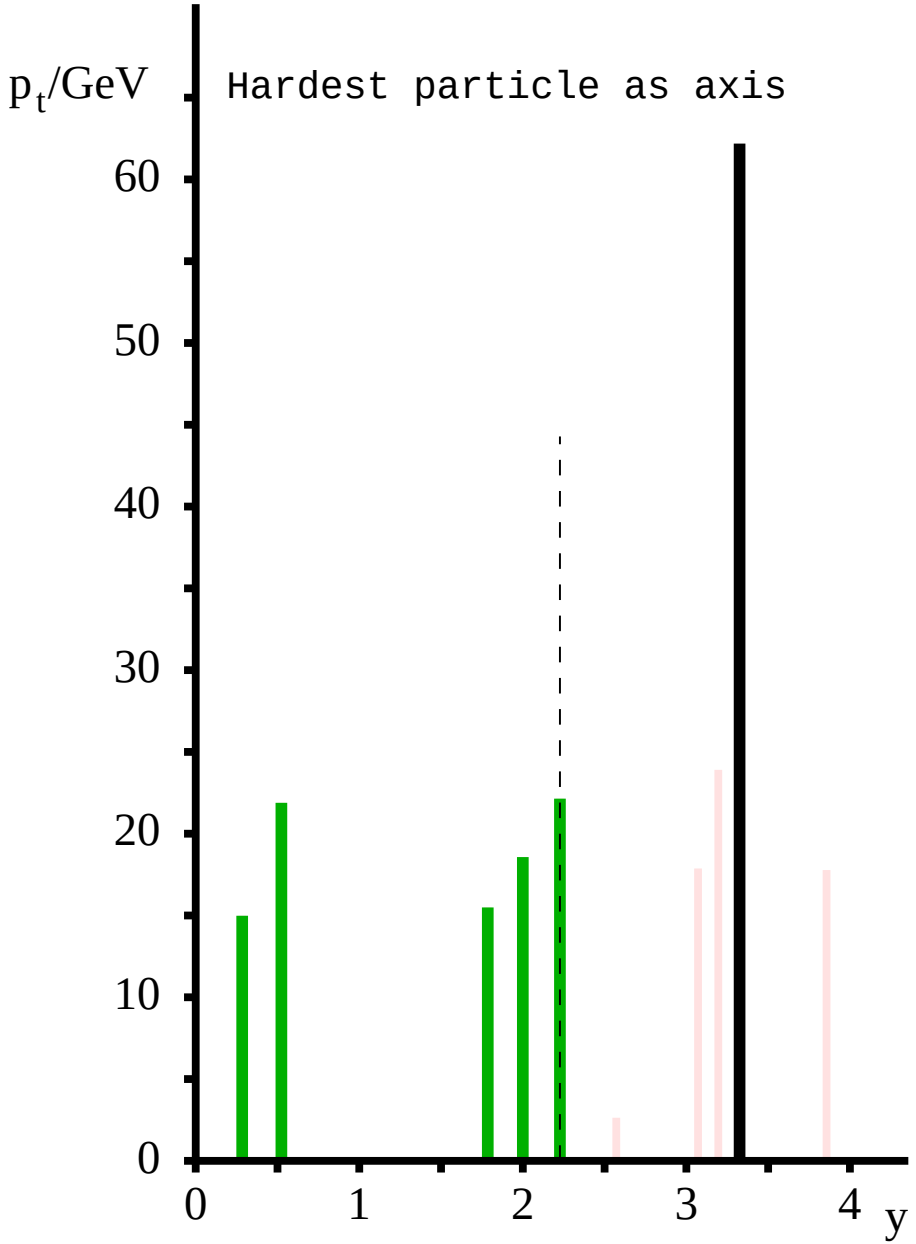
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ⇒ more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

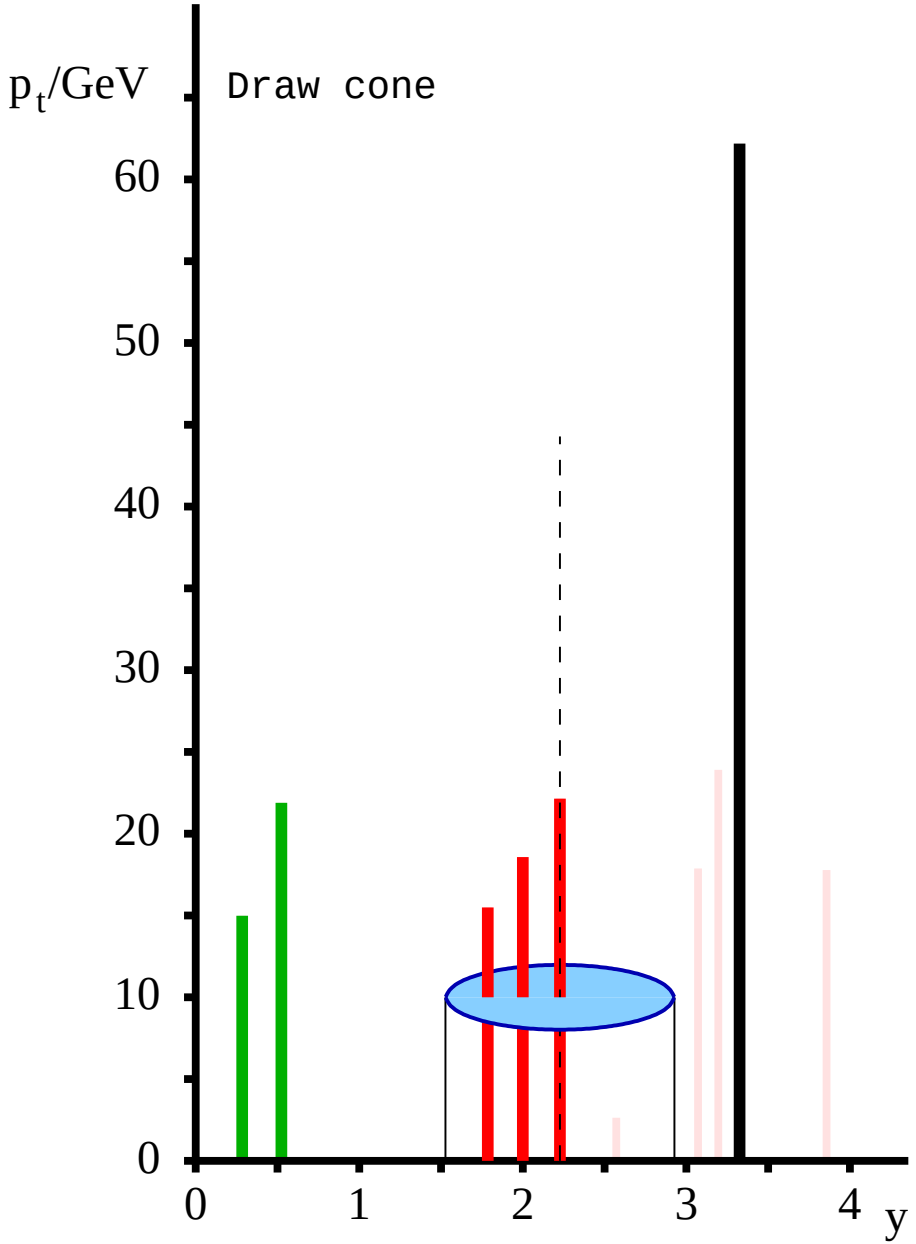
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ⇒ more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

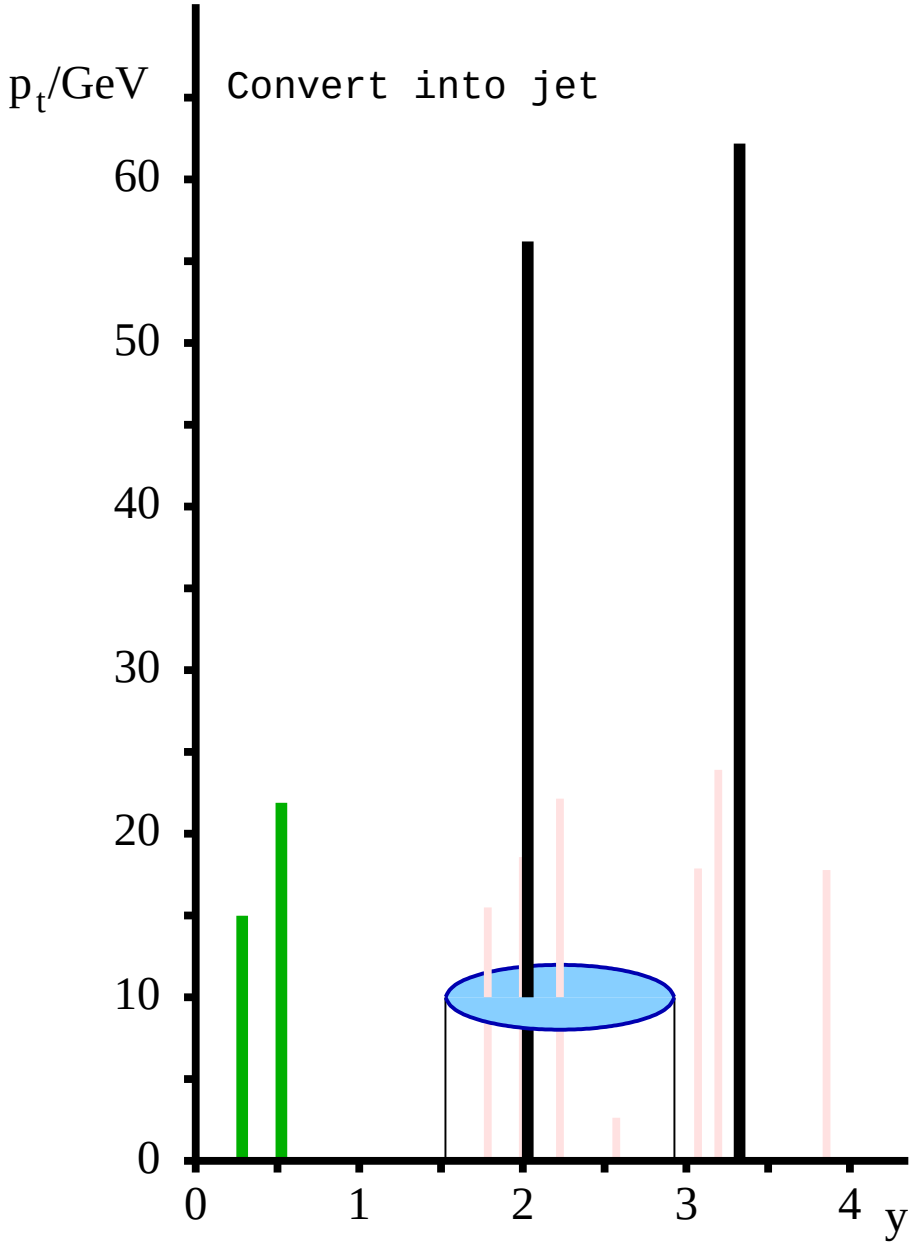
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ⇒ more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

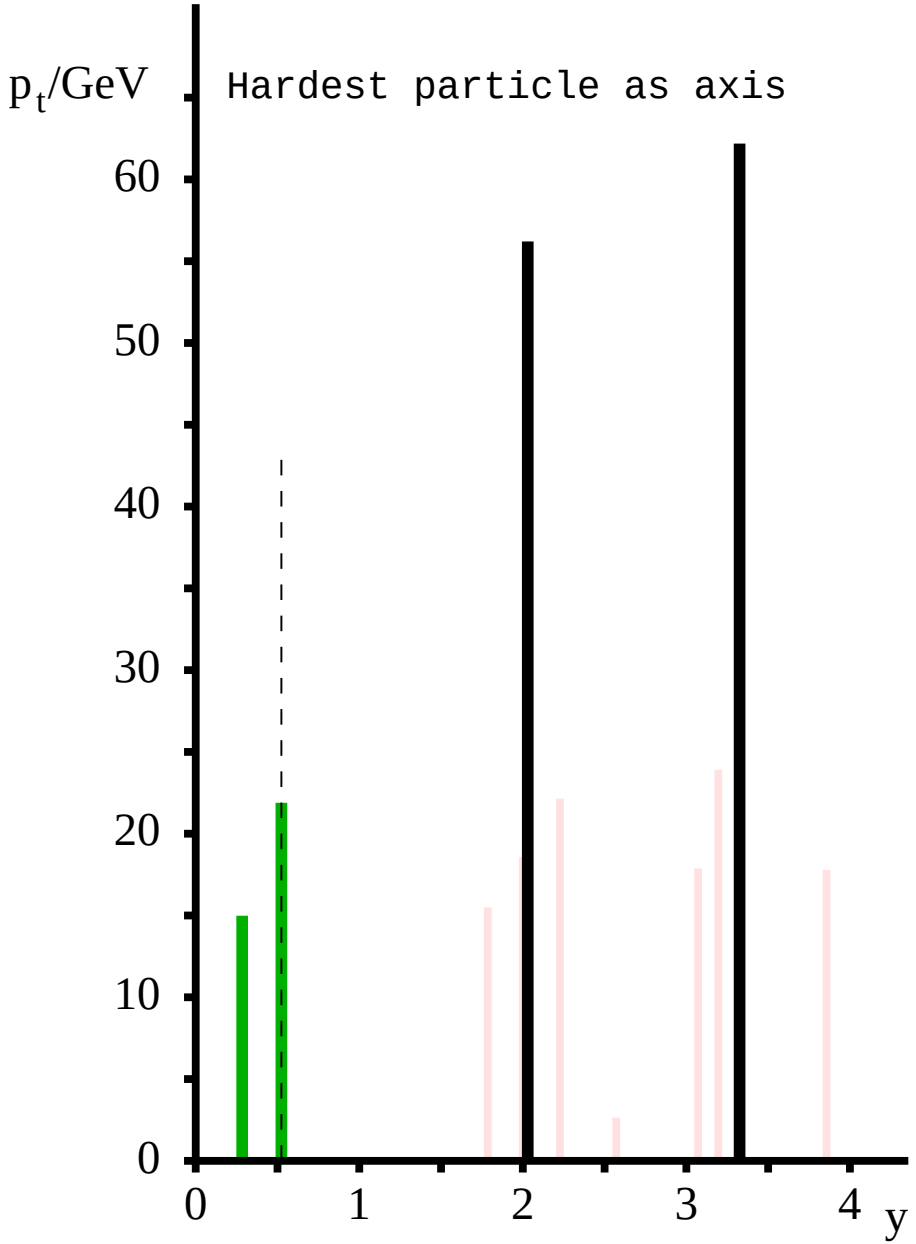
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe ⇒ more later...
- Cone and seed axis may not coincide ⇒ iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

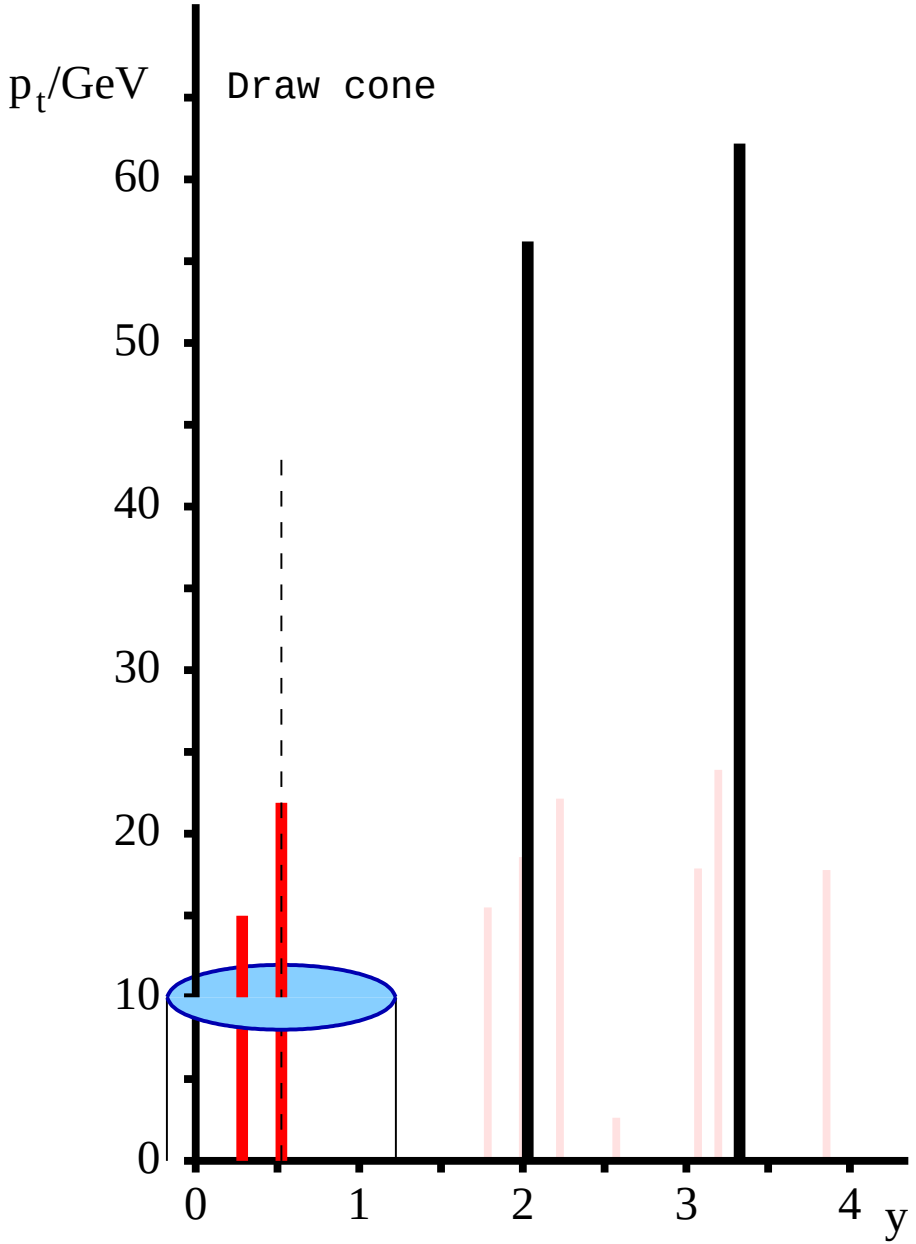
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe — more later...
- Cone and seed axis may not coincide → iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

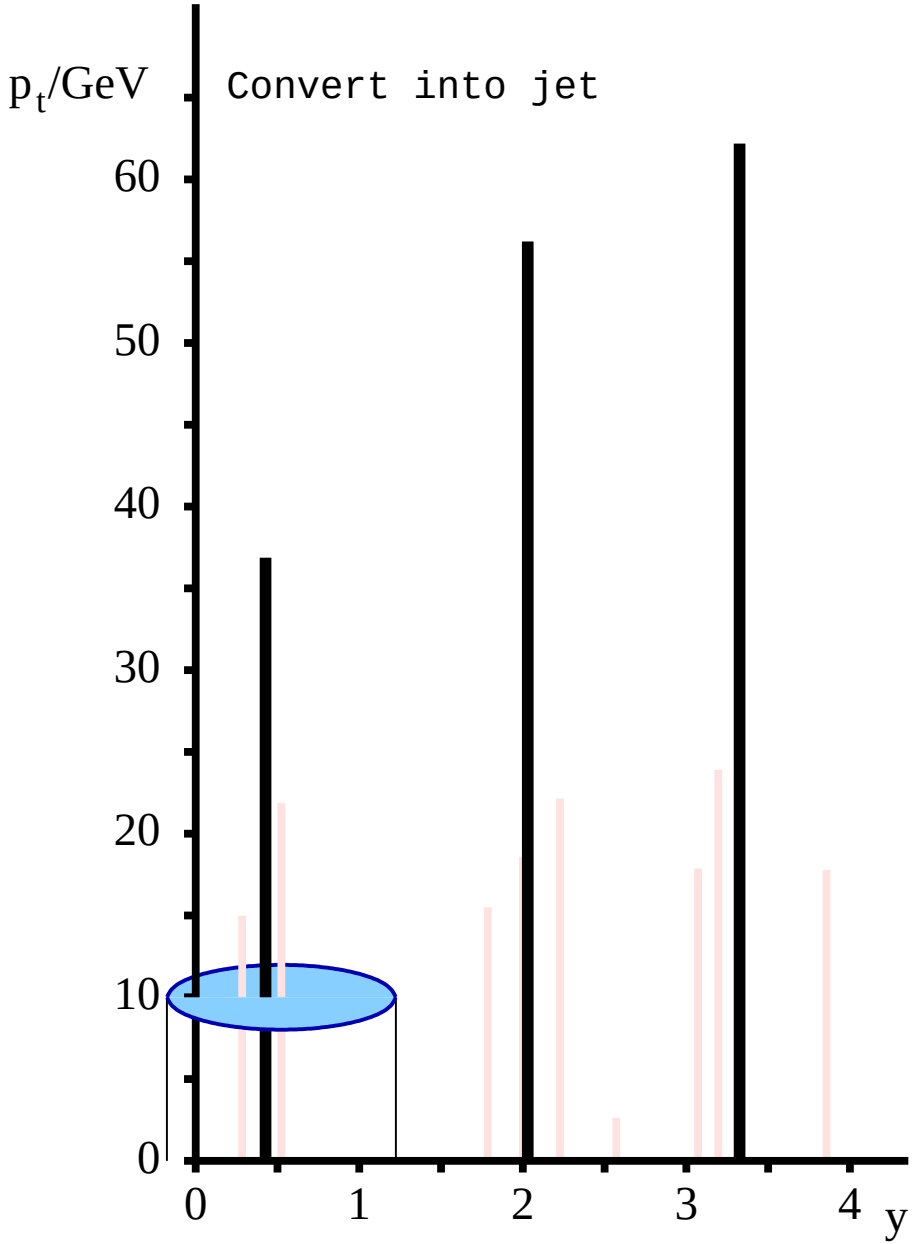
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe — more later...
- Cone and seed axis may not coincide → iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

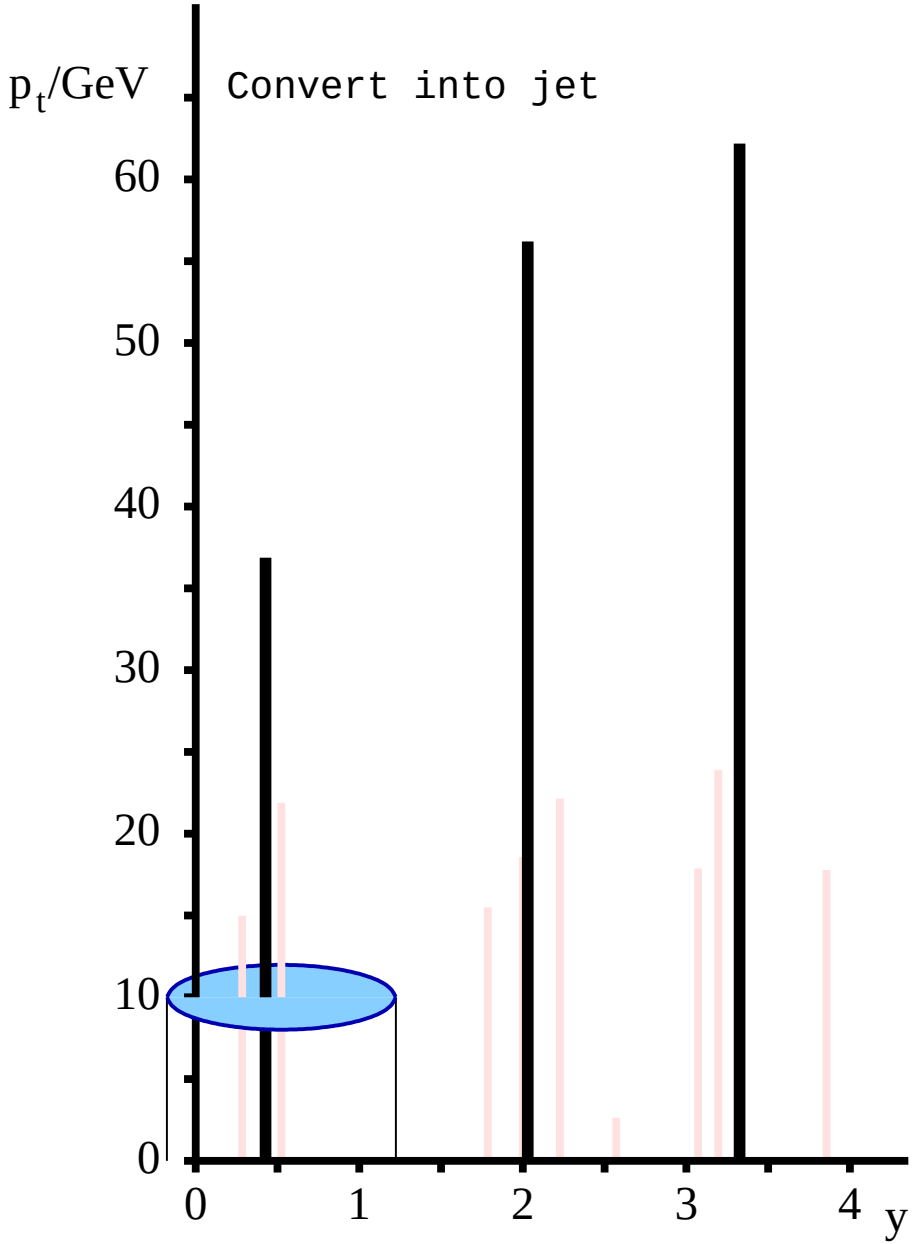
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe — more later...
- Cone and seed axis may not coincide → iteration

Fixed Cone, Prog Removal (FC-PR)



The simplest of the cones

PyCell, CellJet, GetJet

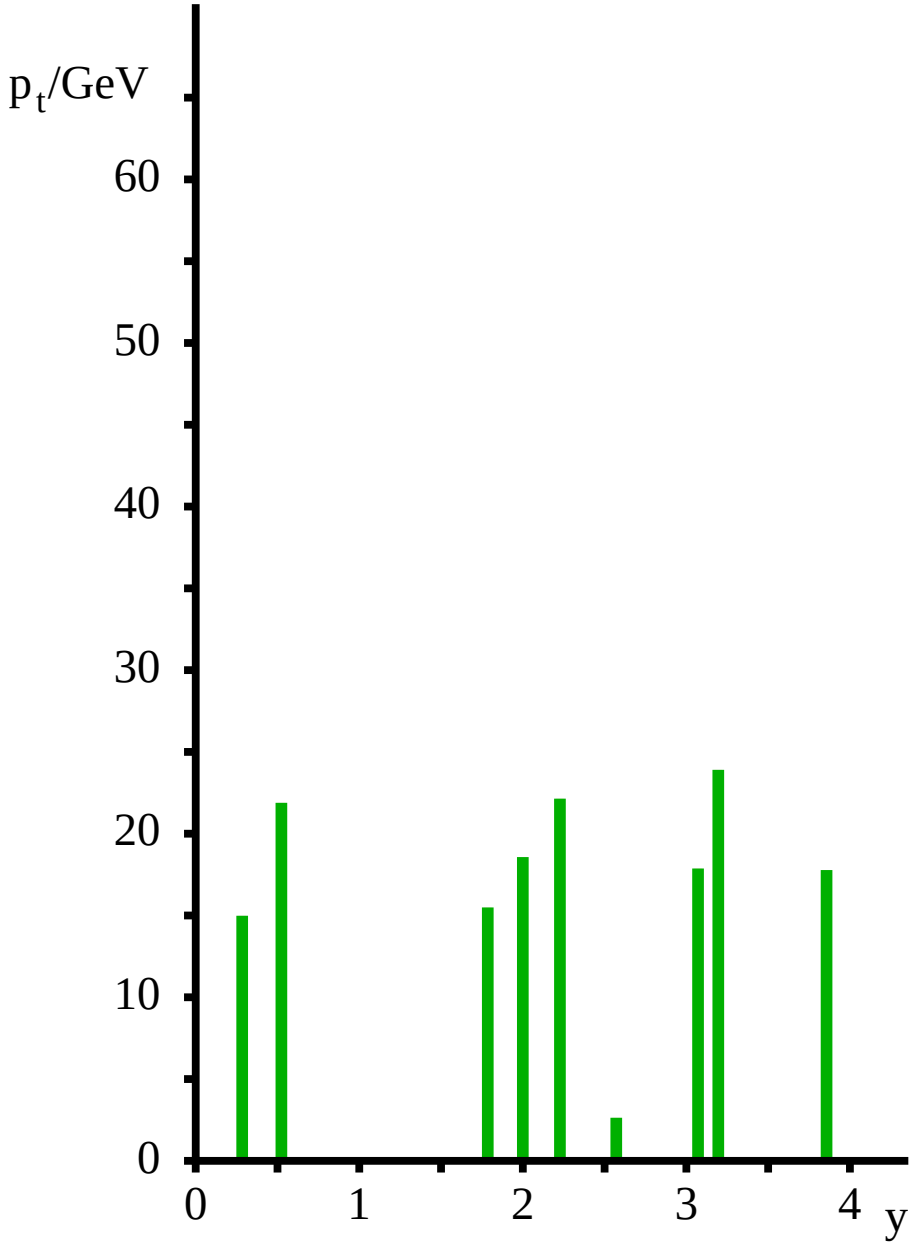
Used e.g. BSM theory; Alpgen MLM

- Take hardest particle as seed for cone axis
- Draw cone around it
- Convert contents into a "jet" and remove them from the event
- Repeat until no particles left

Notes

- "Hardest particle" is collinear unsafe — more later...
- Cone and seed axis may not coincide → iteration

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

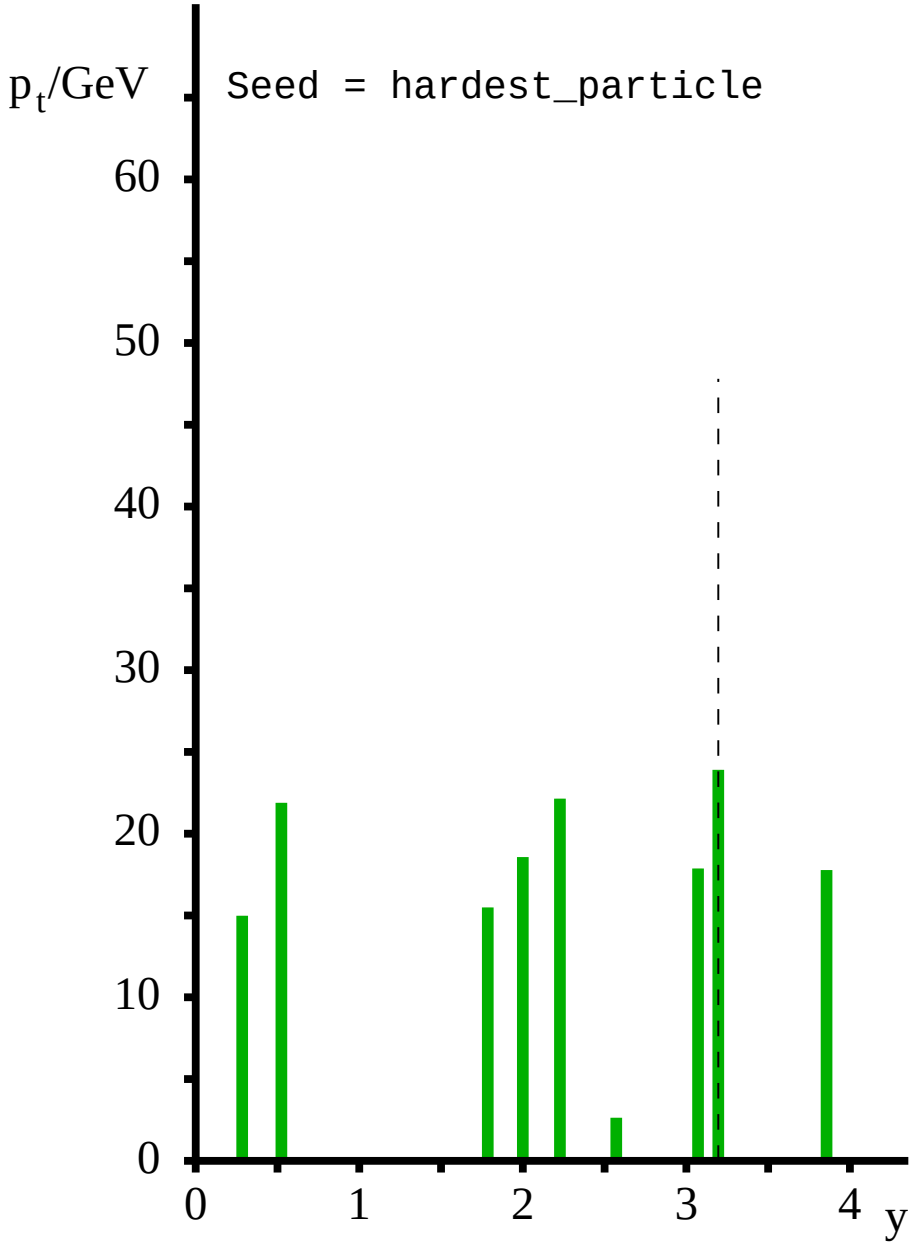
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe — more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

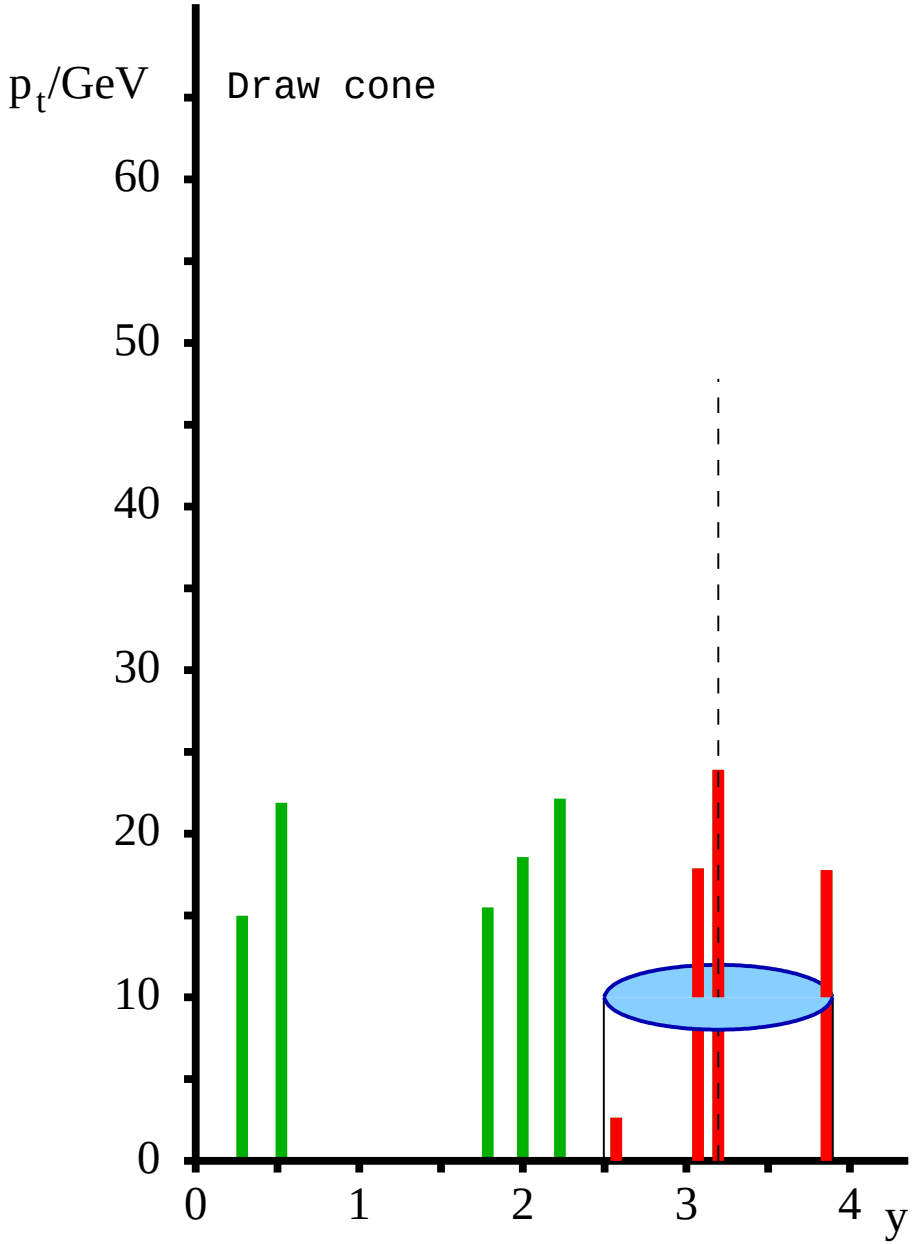
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe — more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

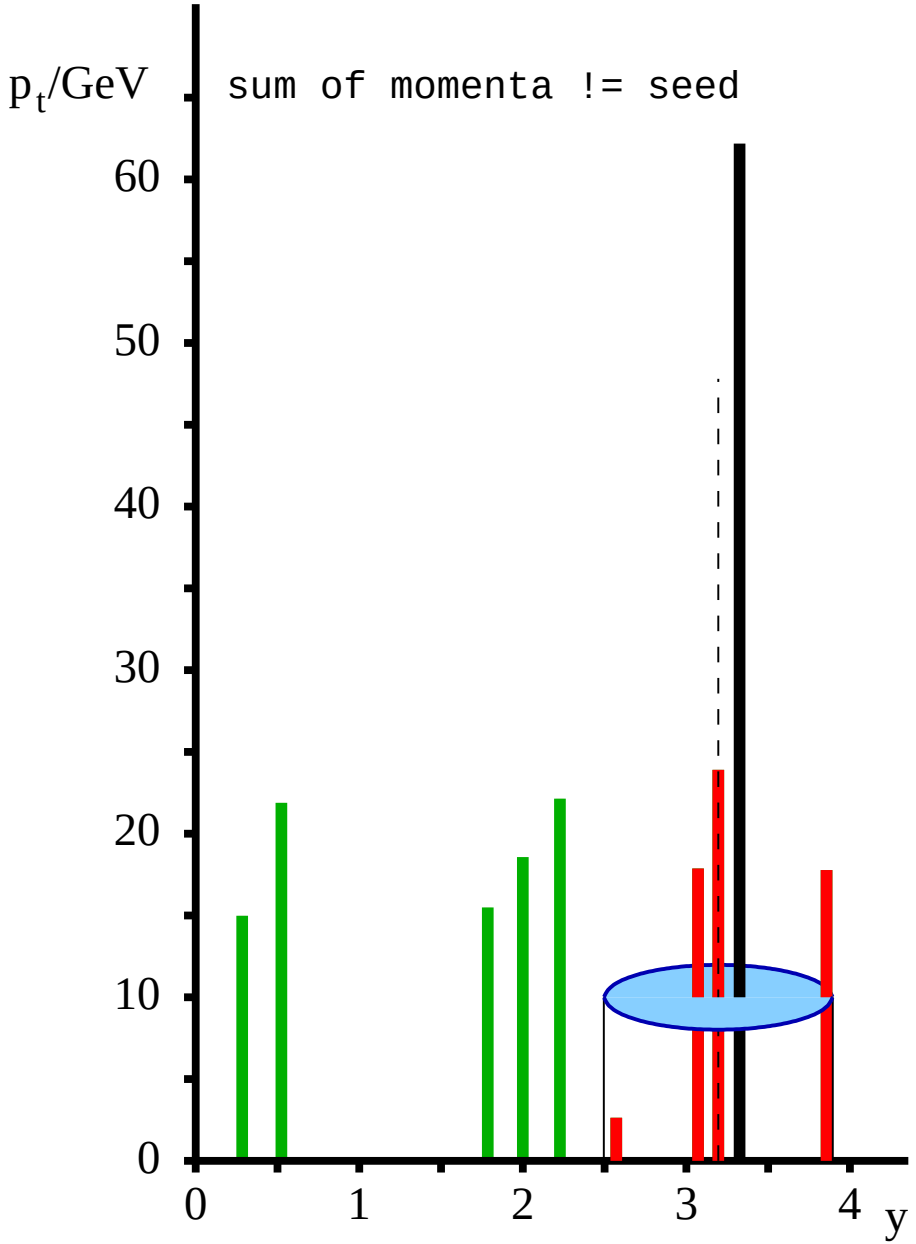
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

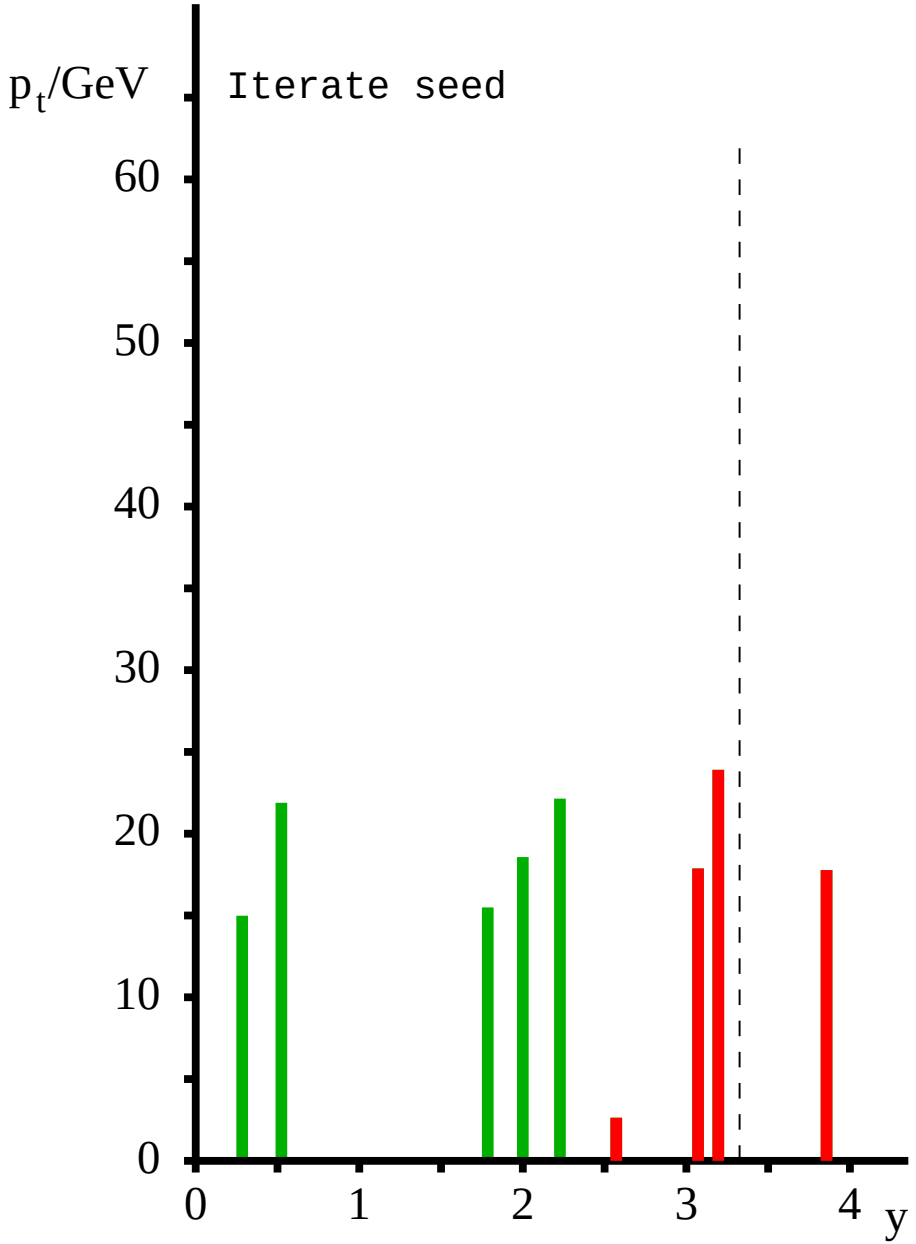
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

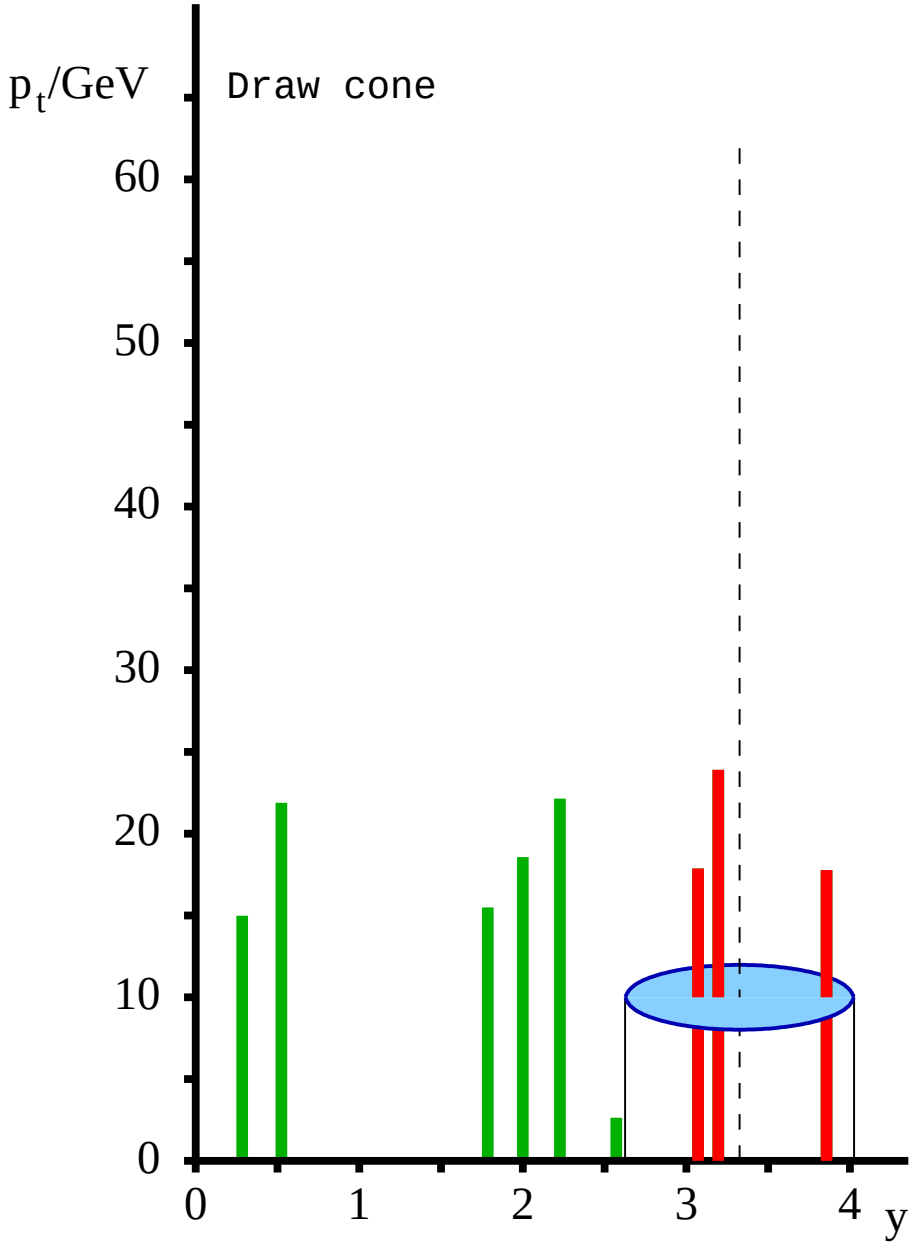
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe — more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

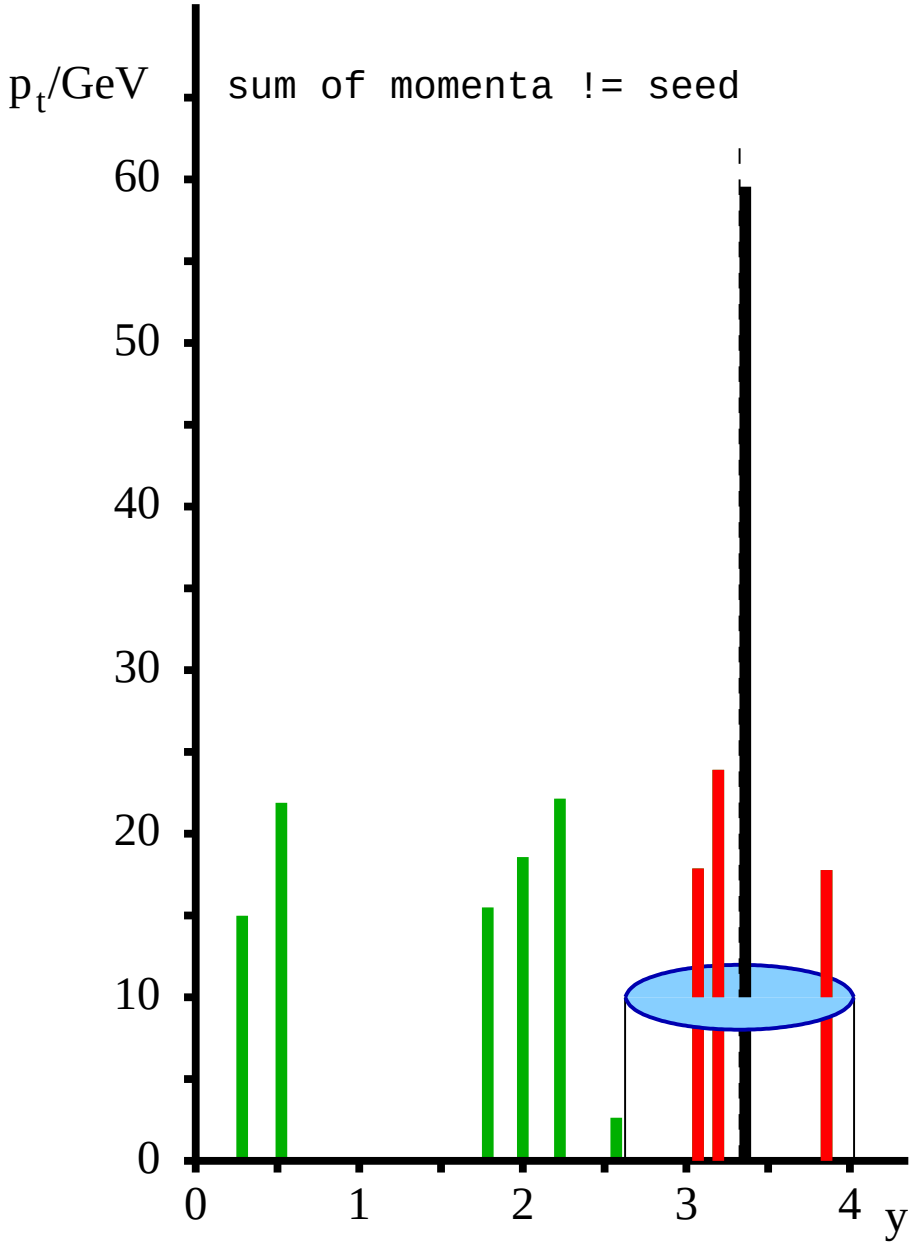
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

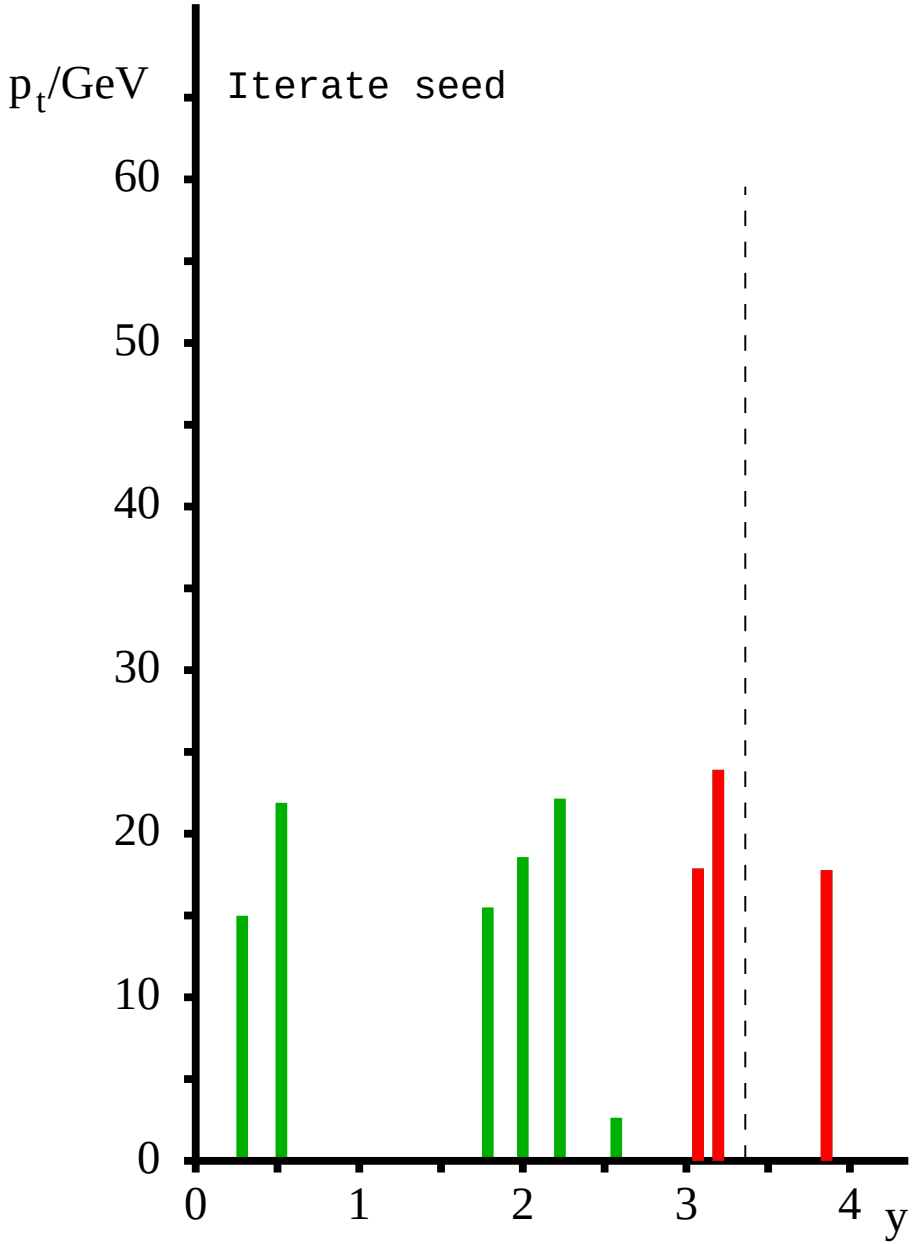
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

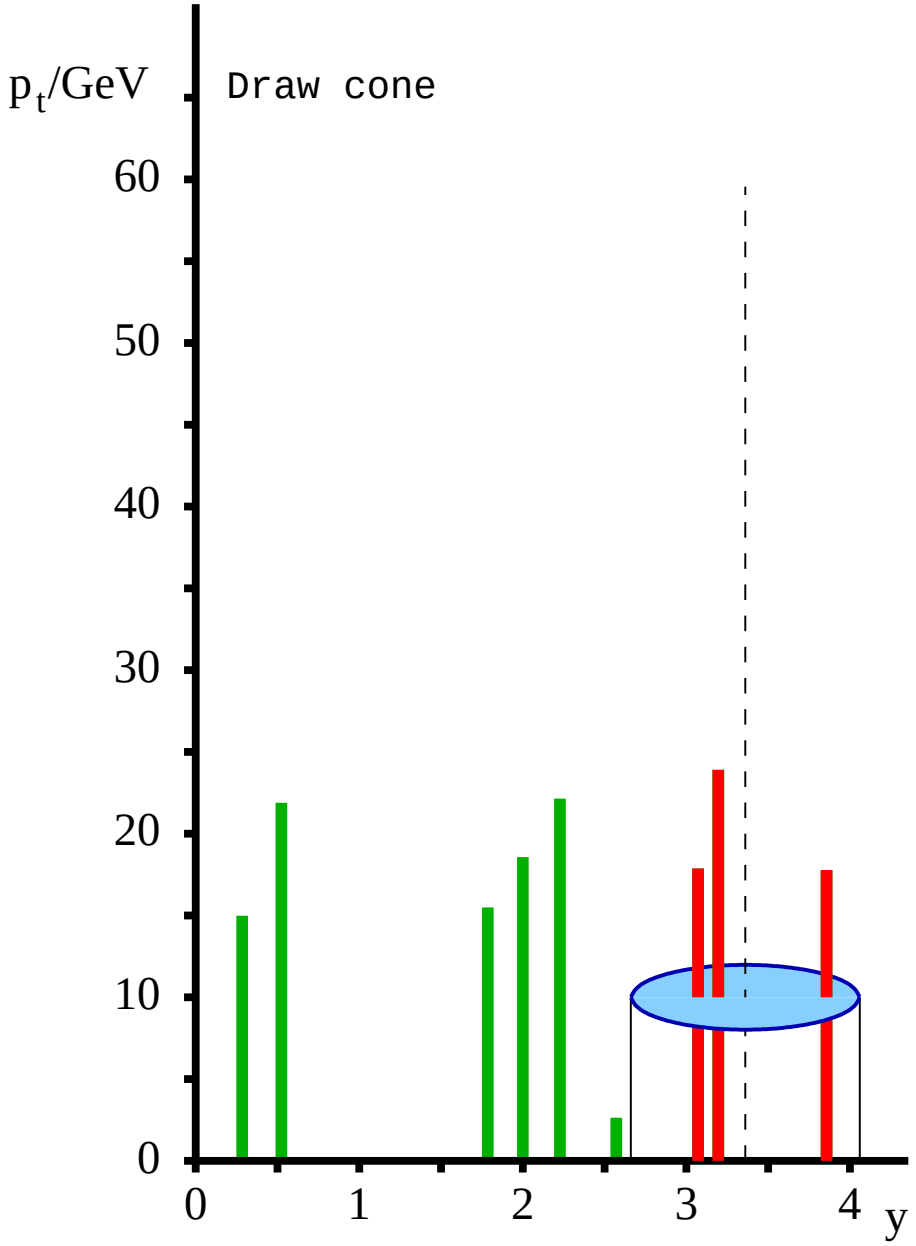
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe … more right away…

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

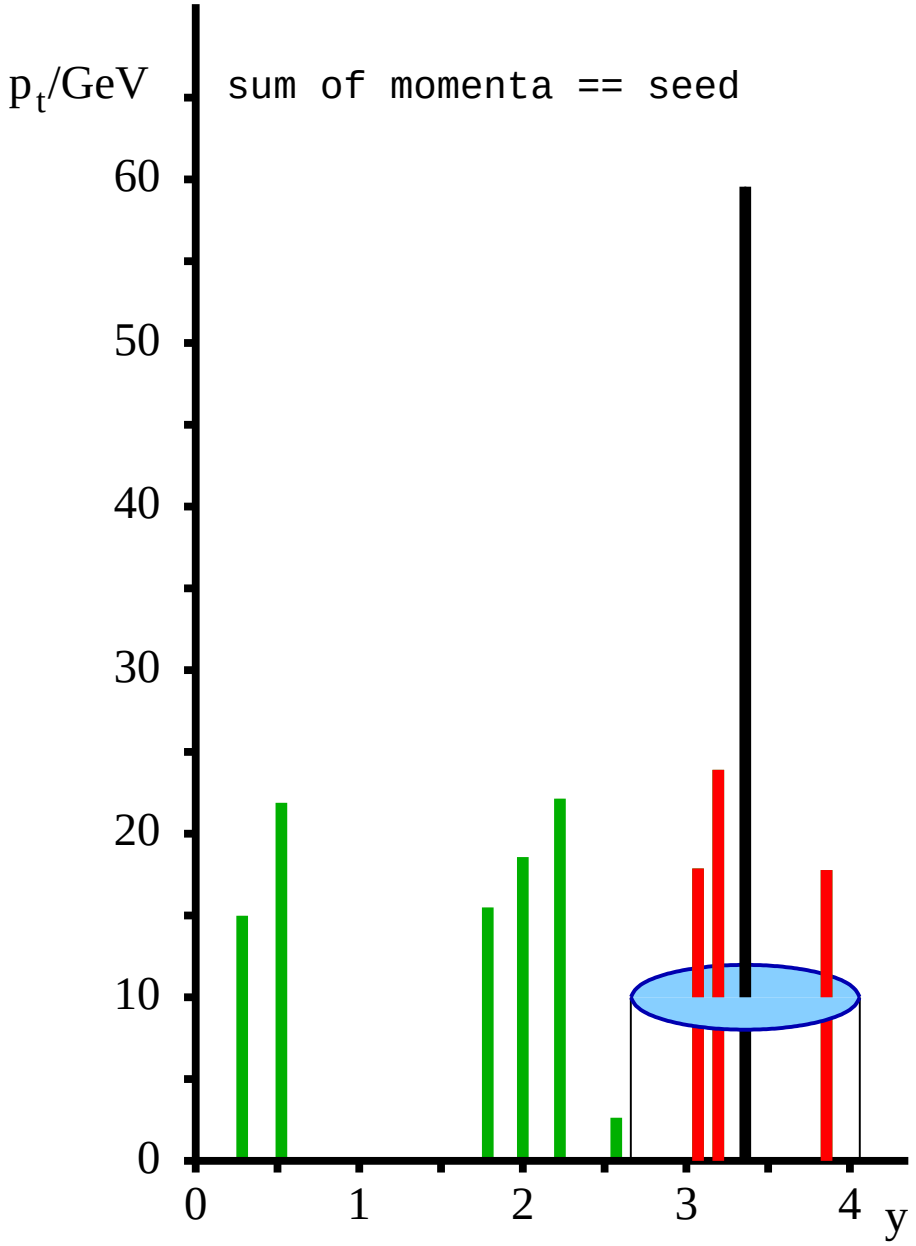
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

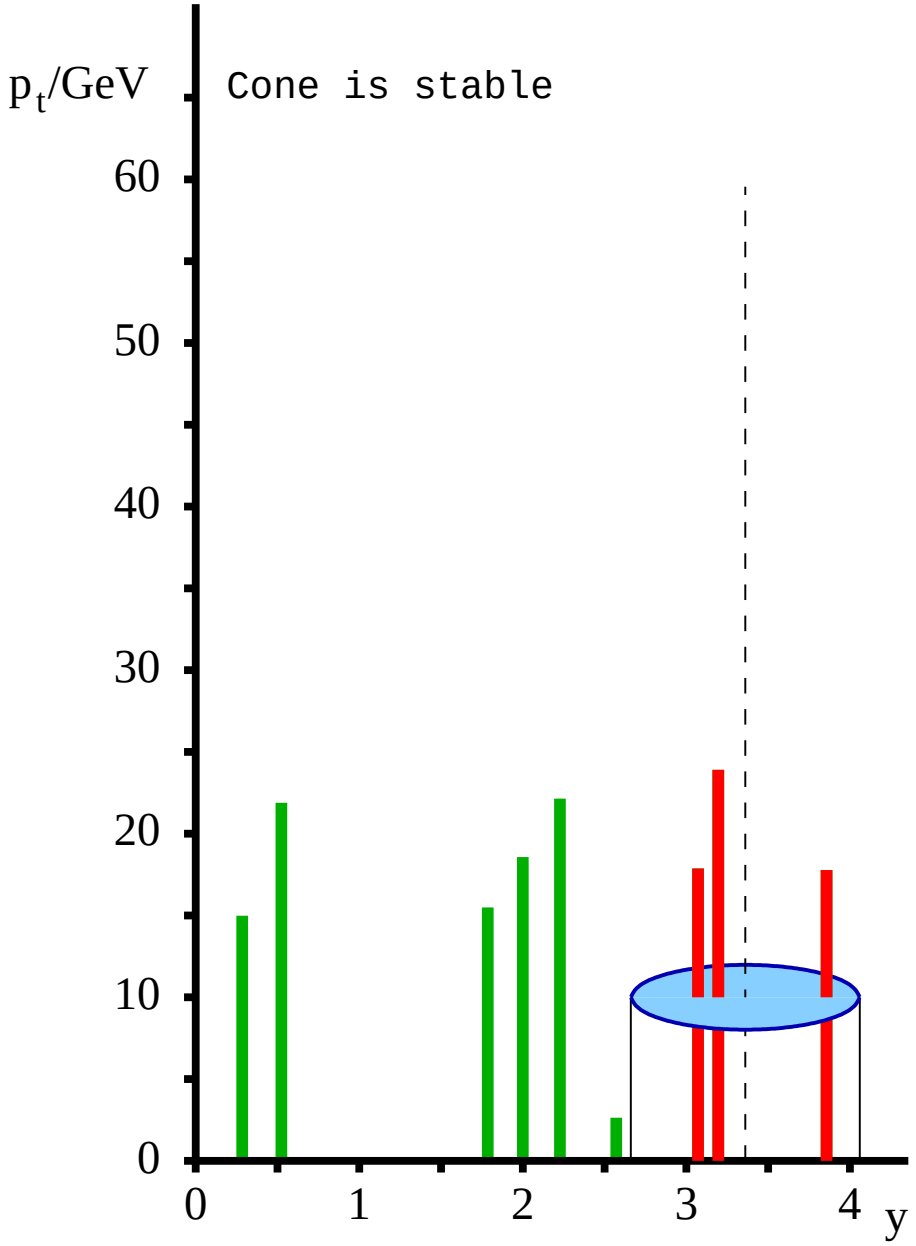
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

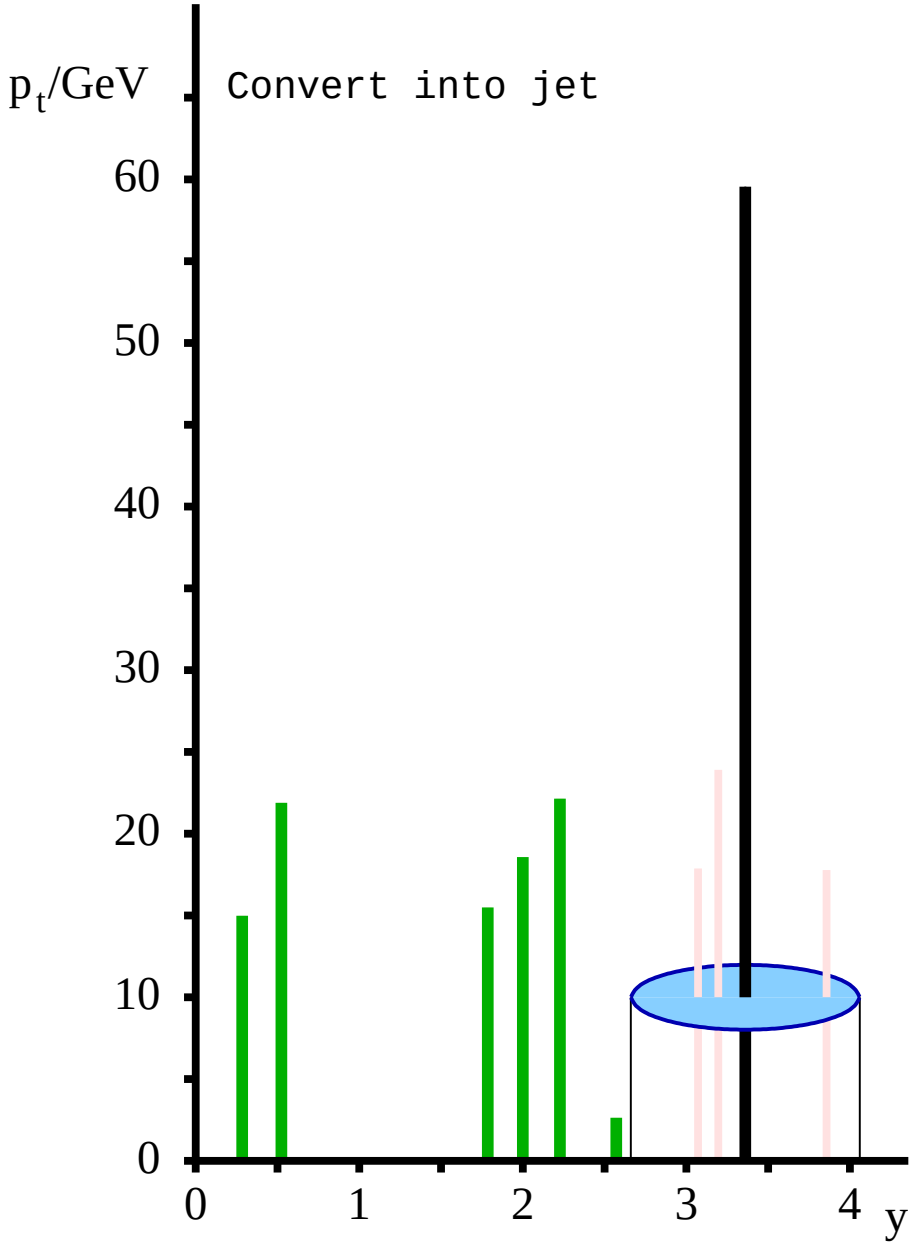
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

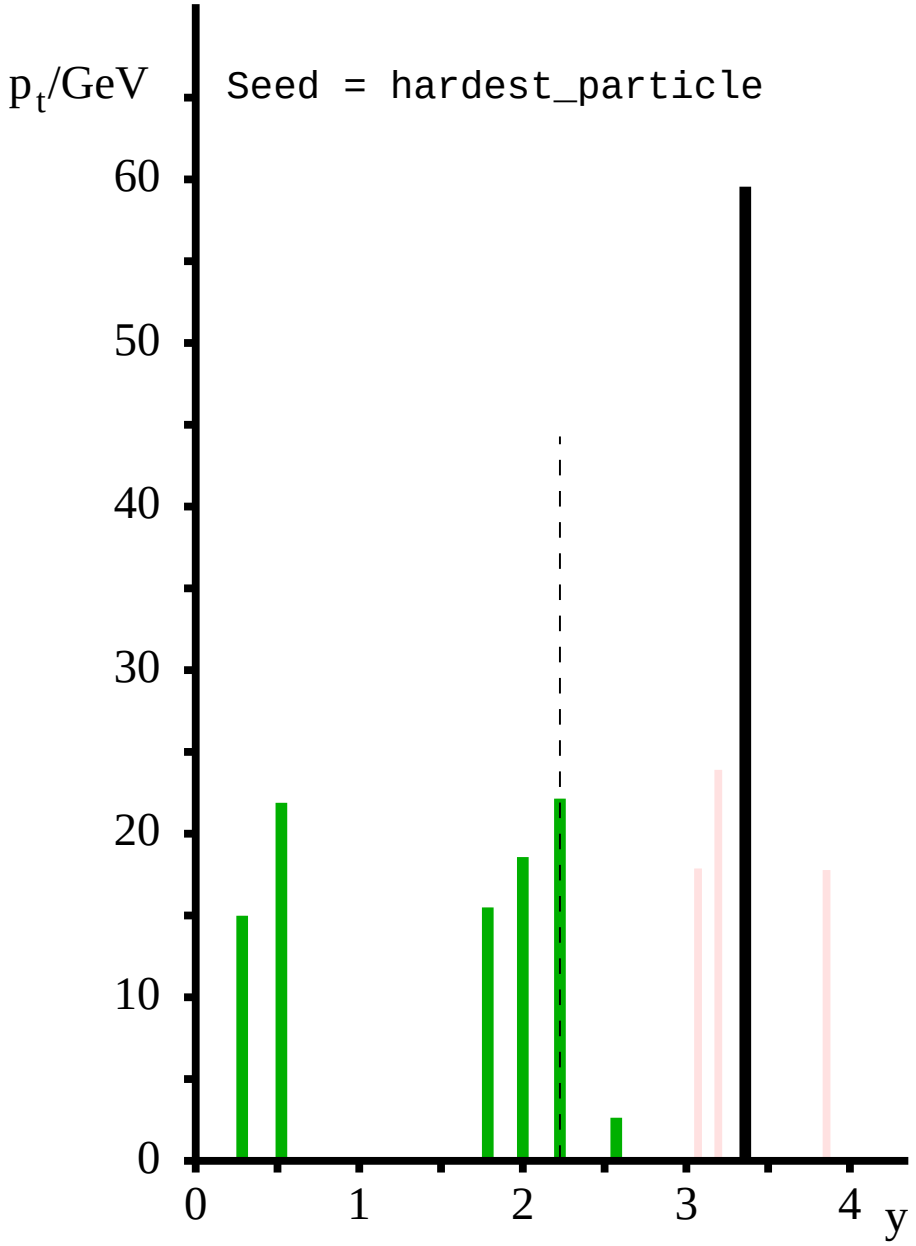
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

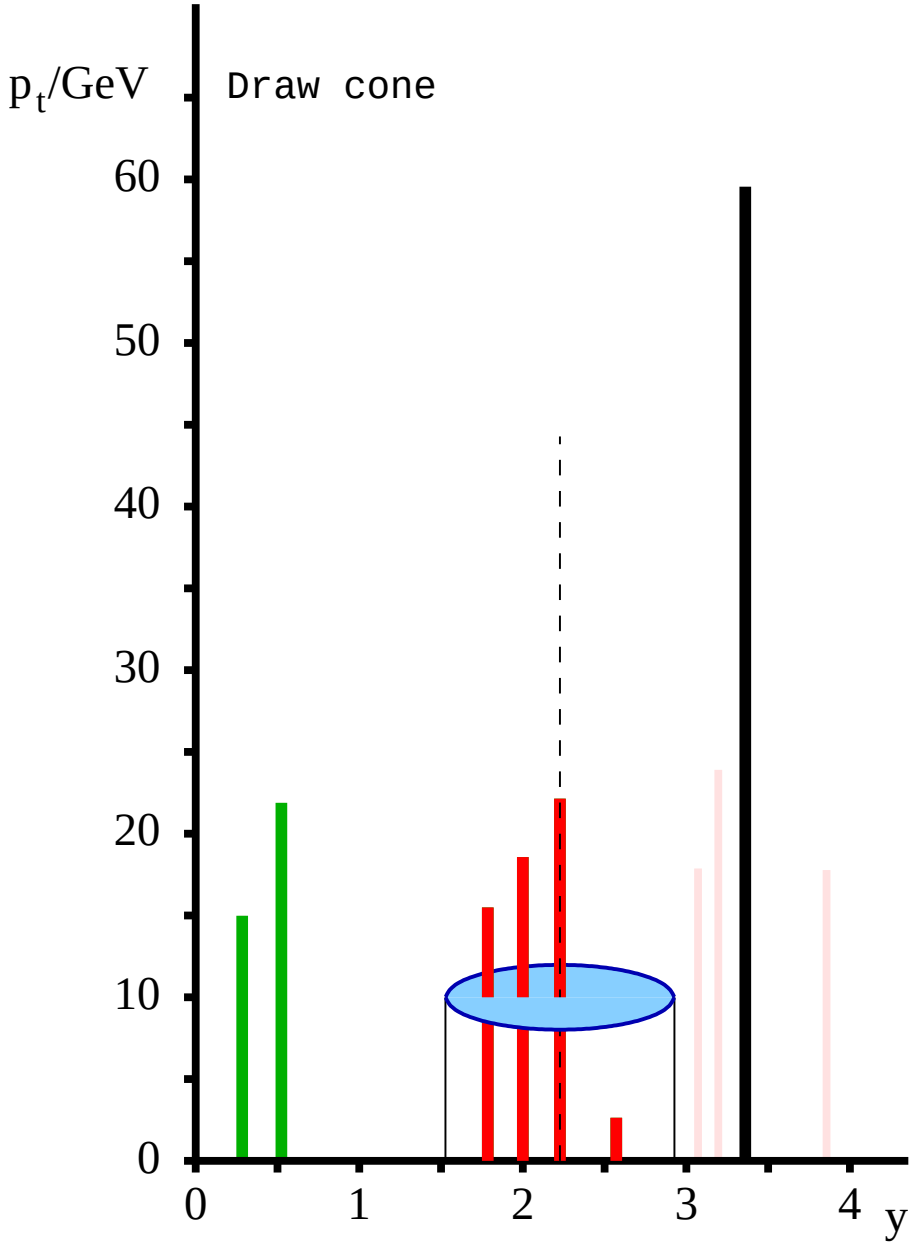
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

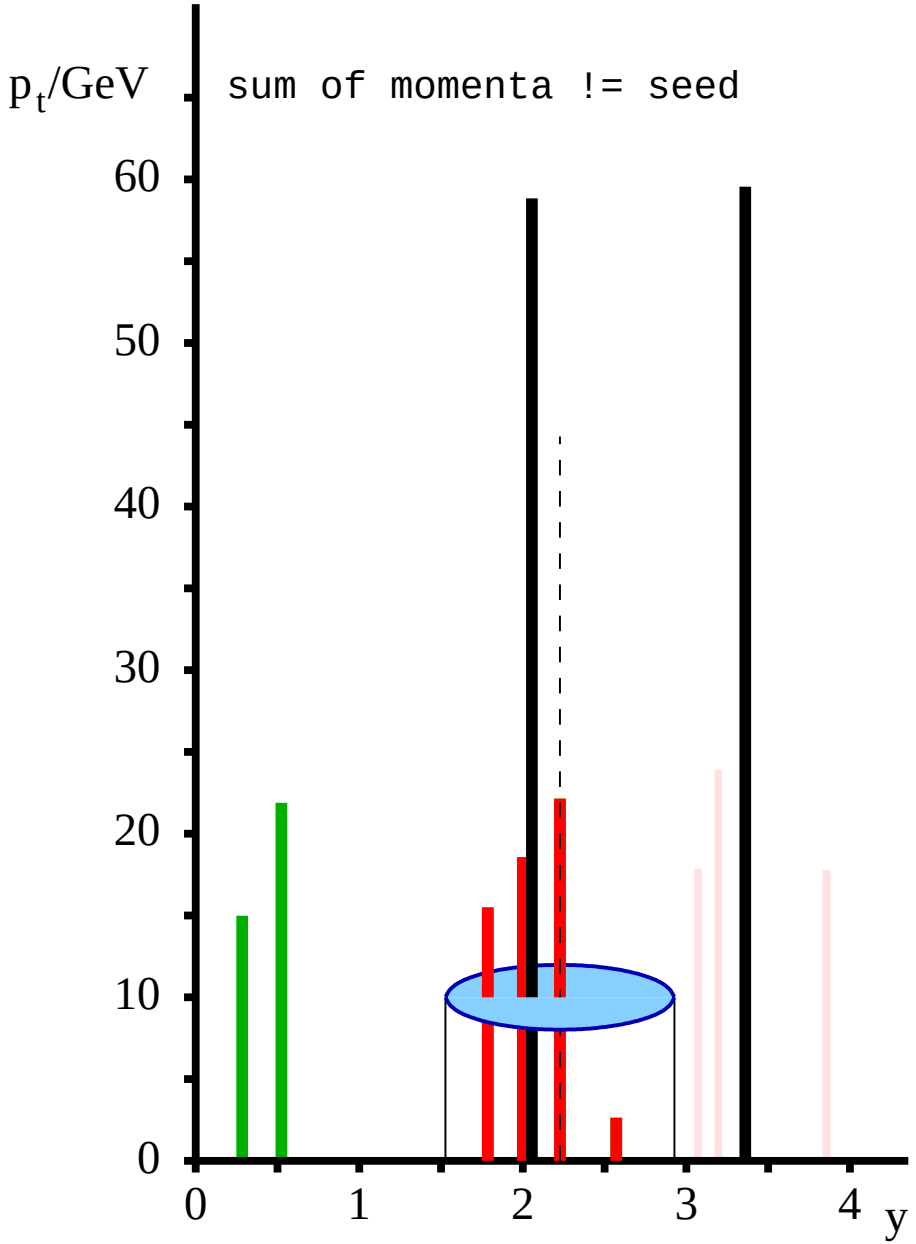
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

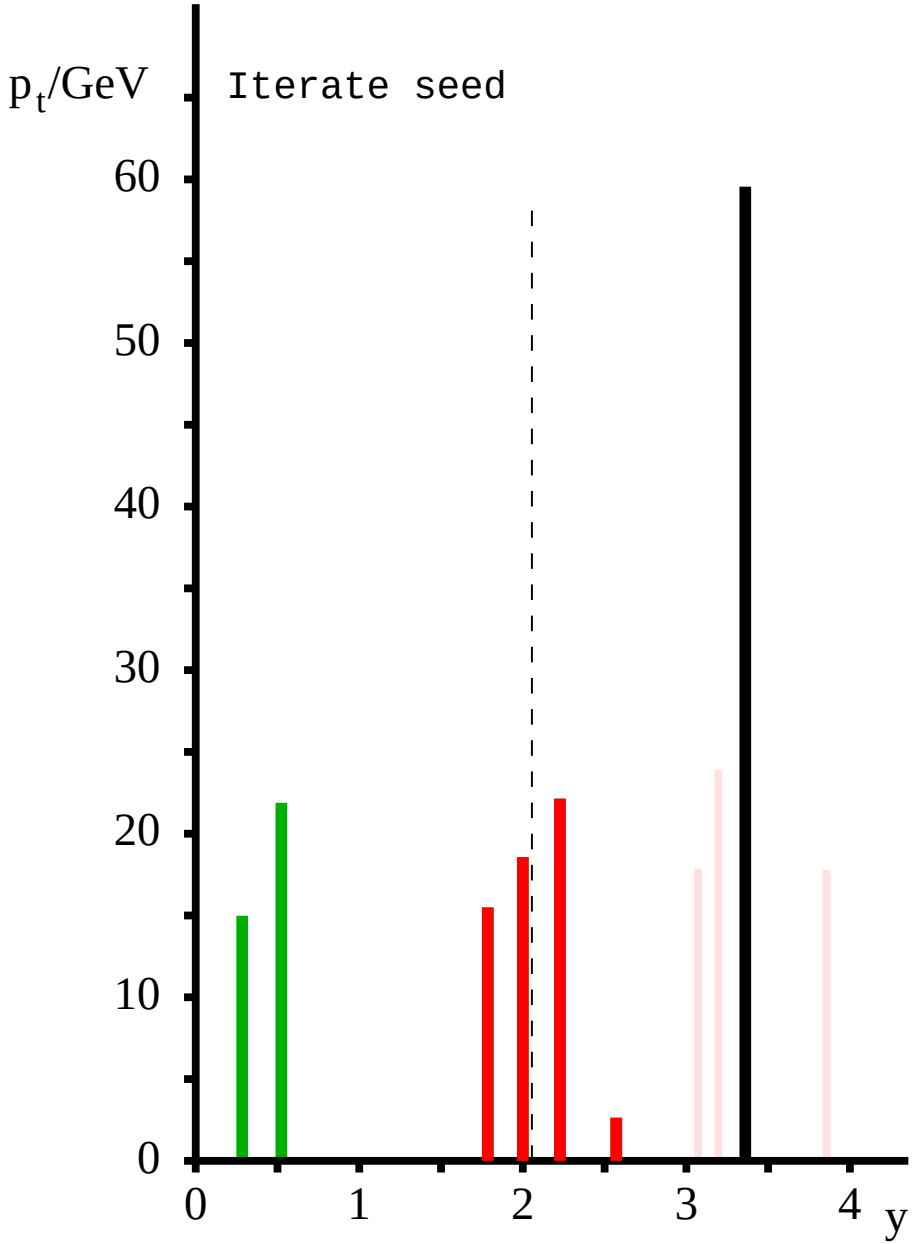
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

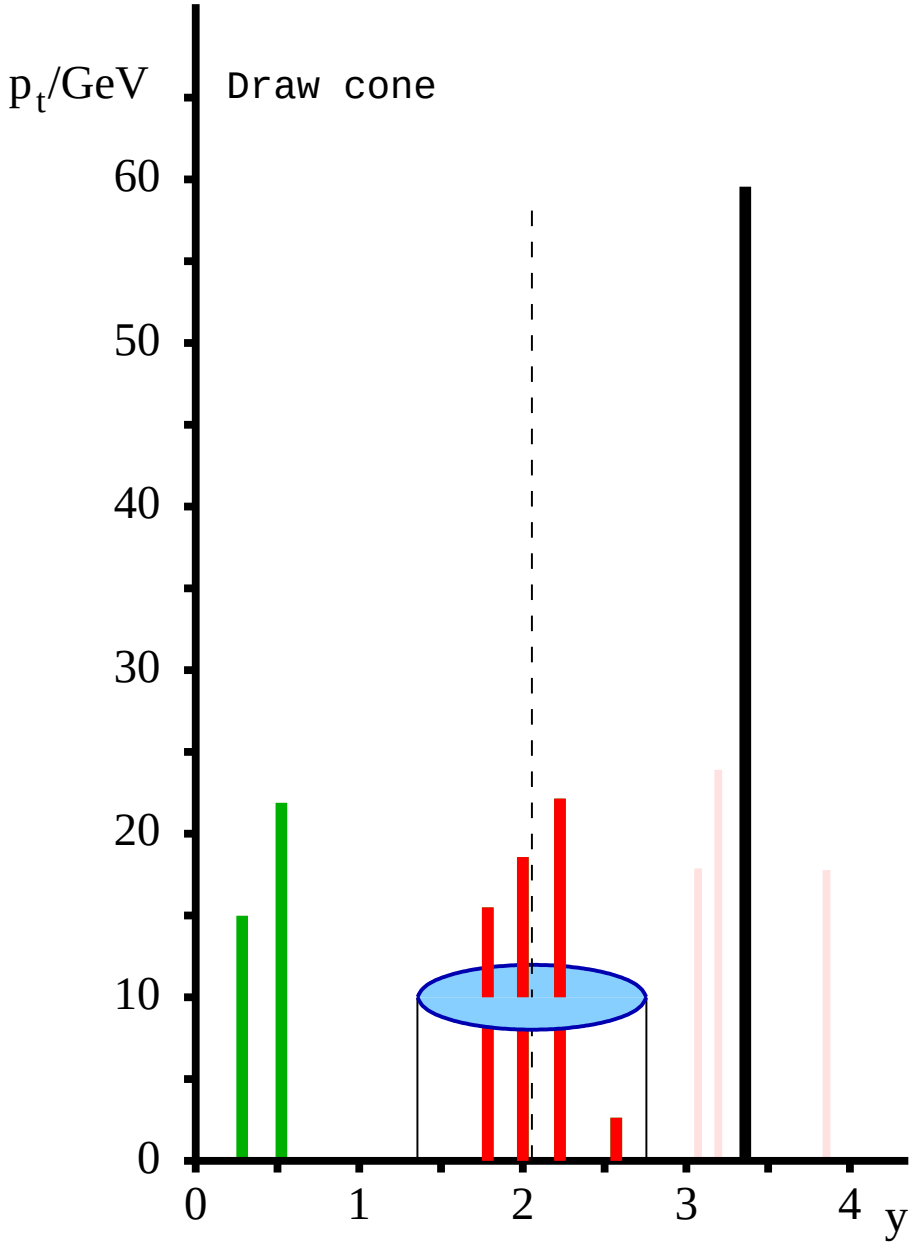
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

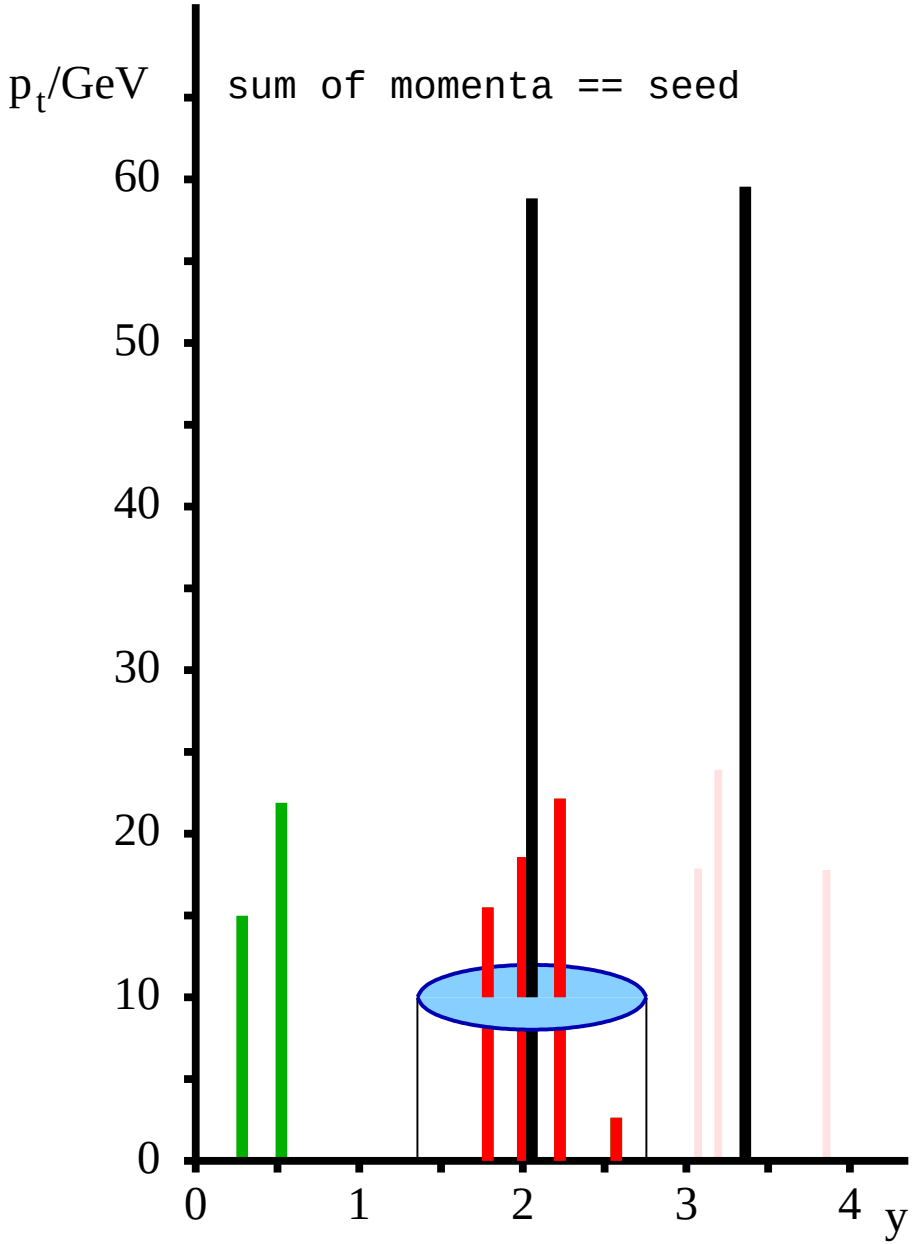
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

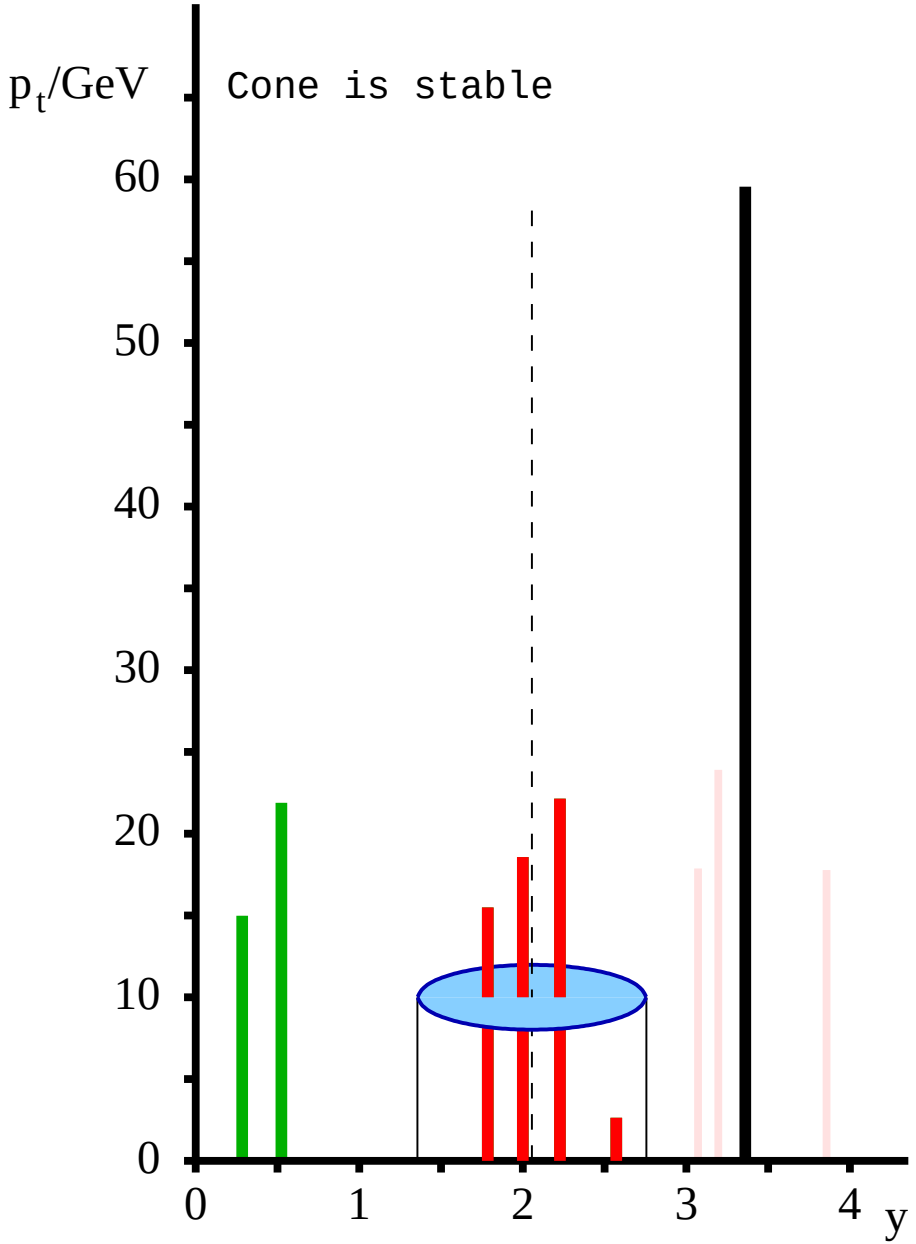
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

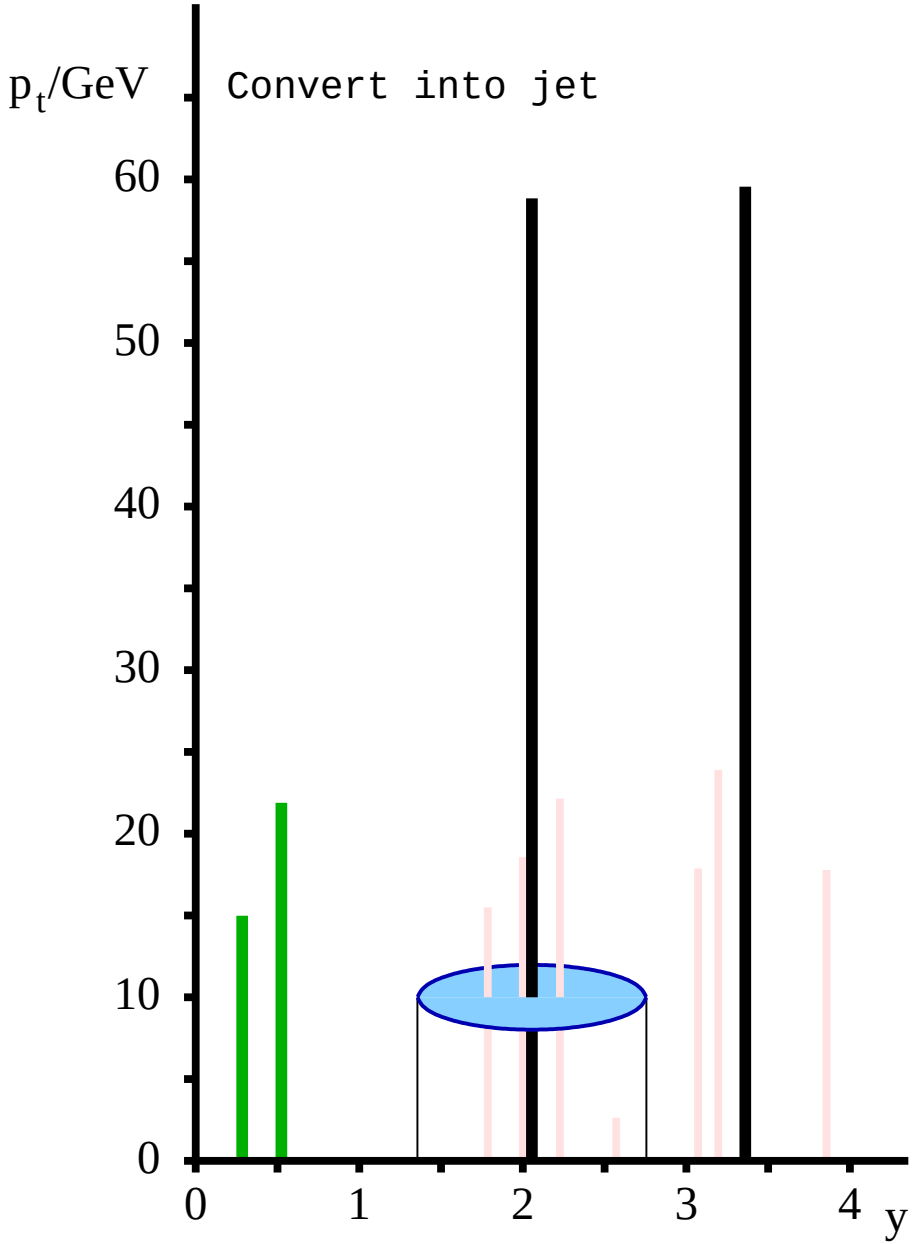
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

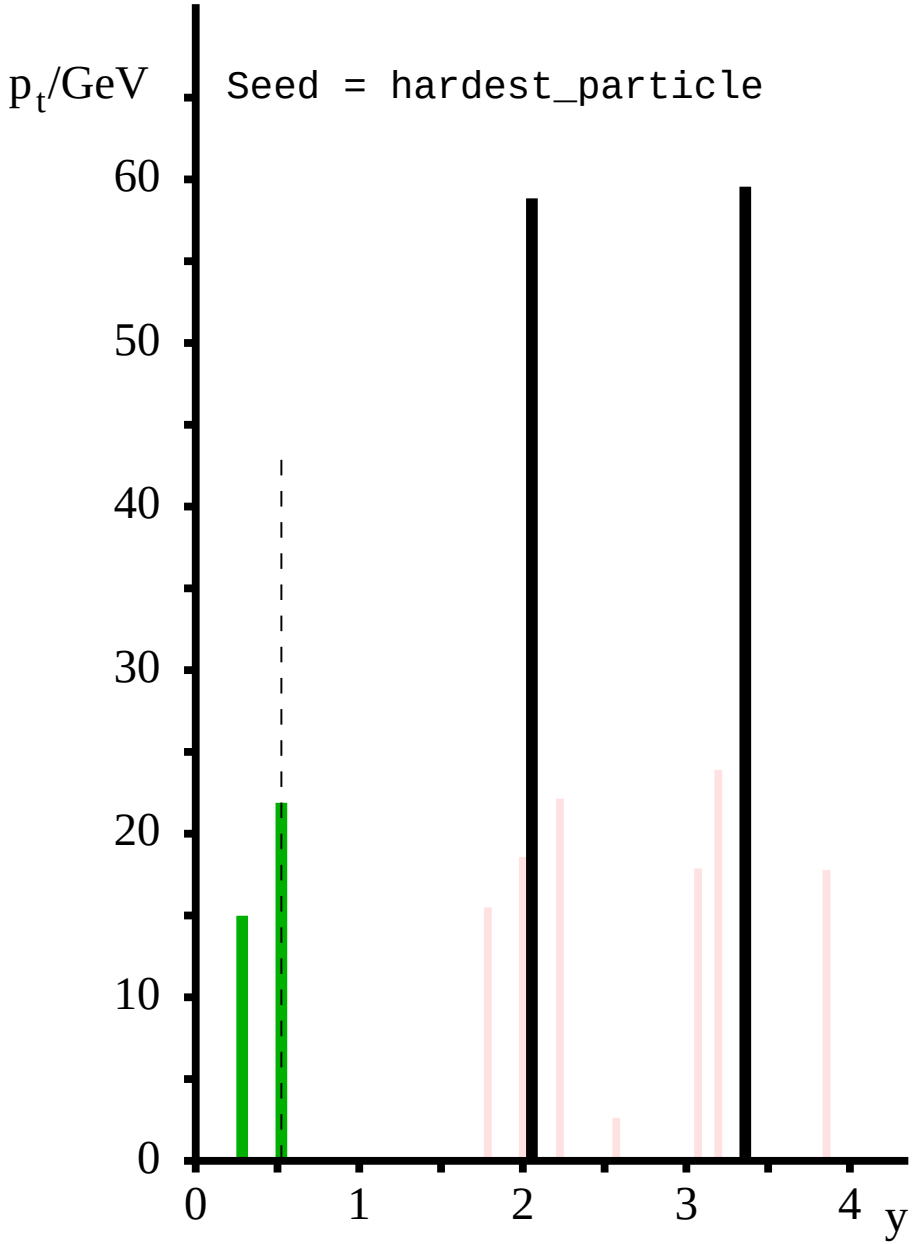
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

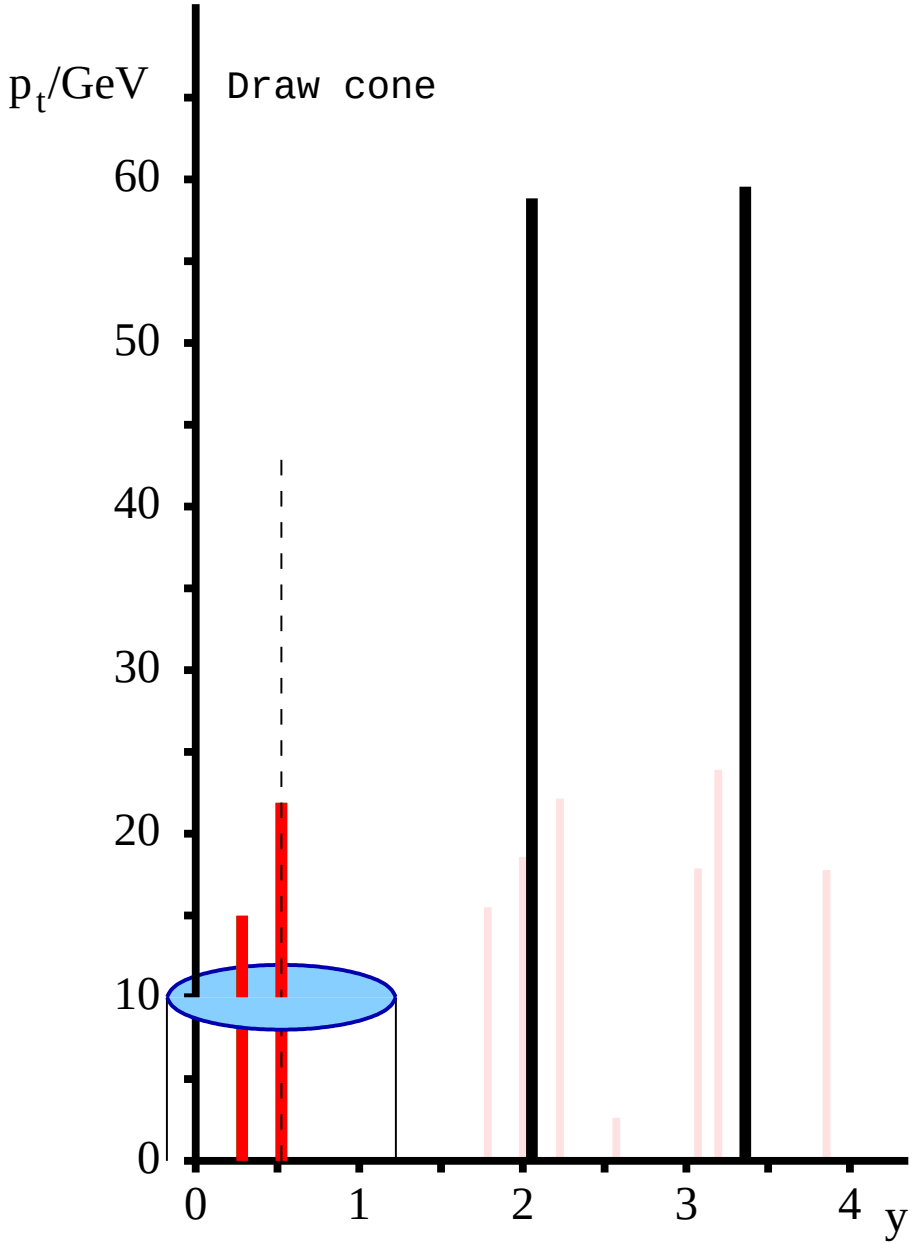
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

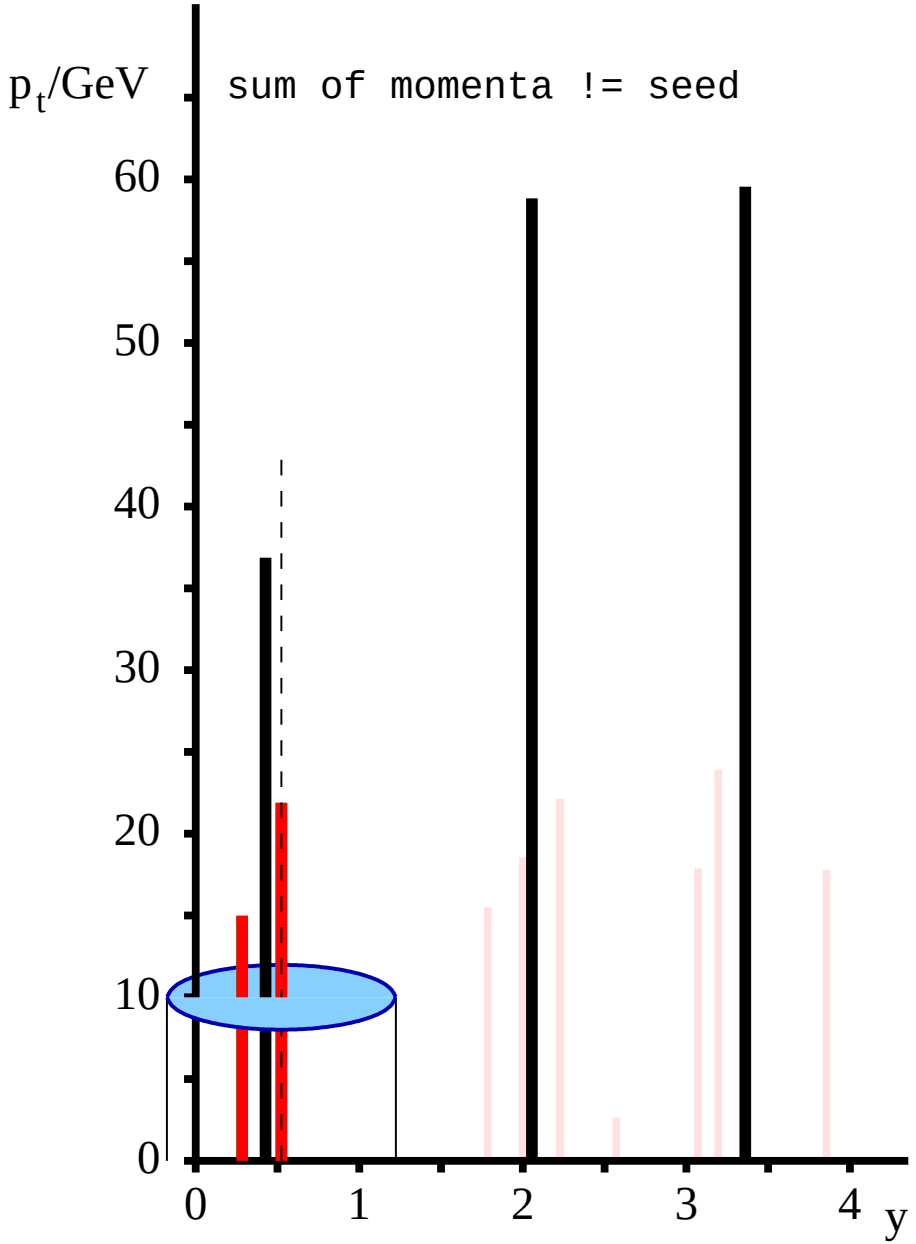
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

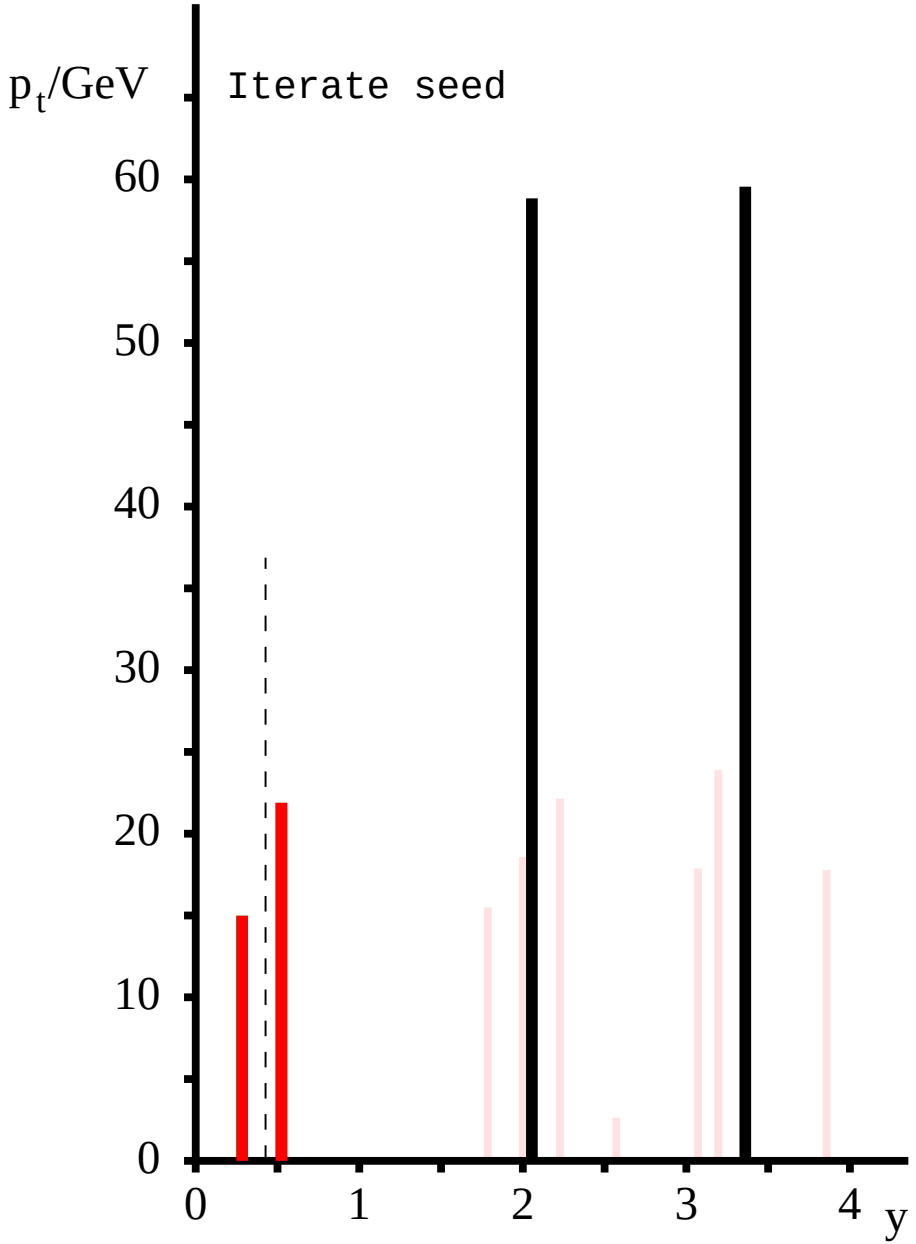
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

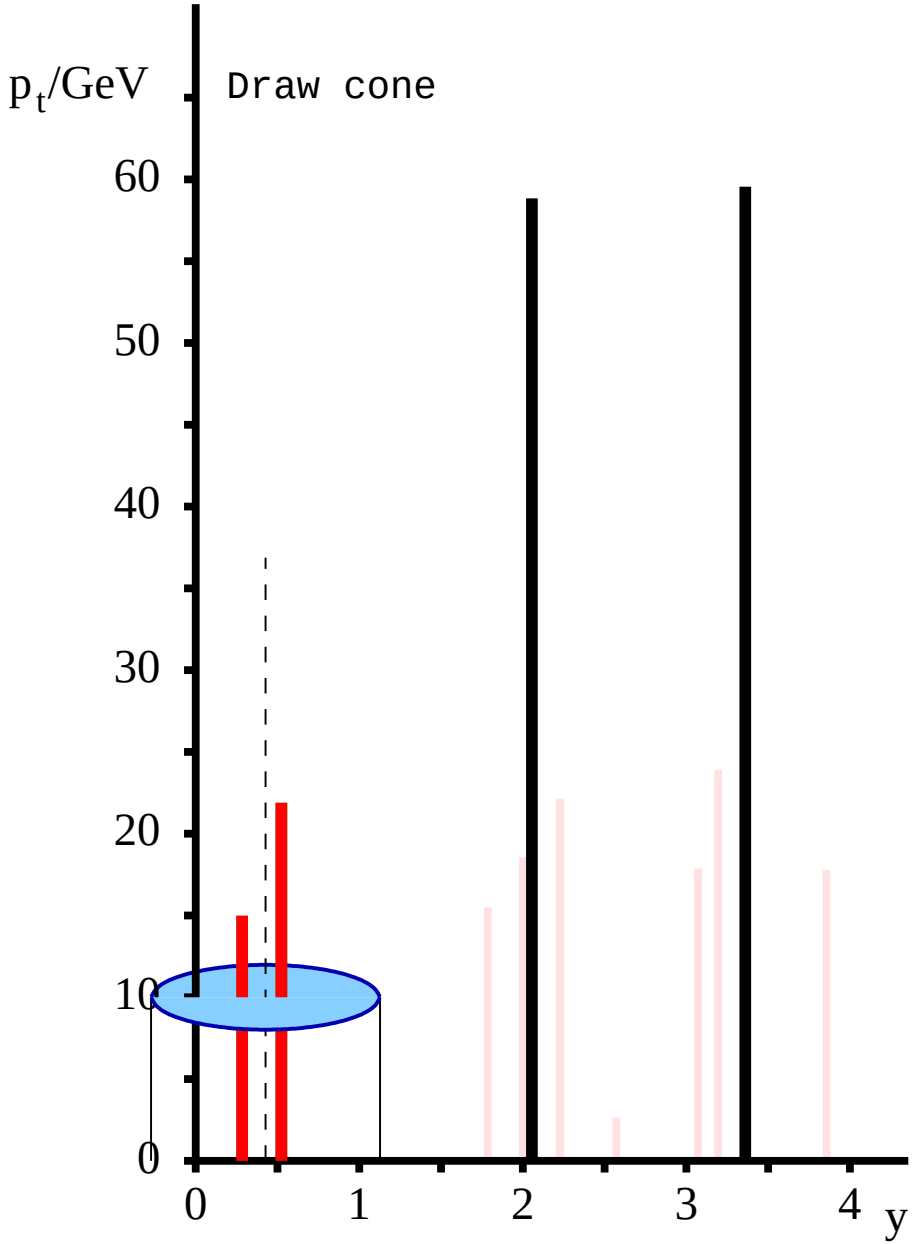
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

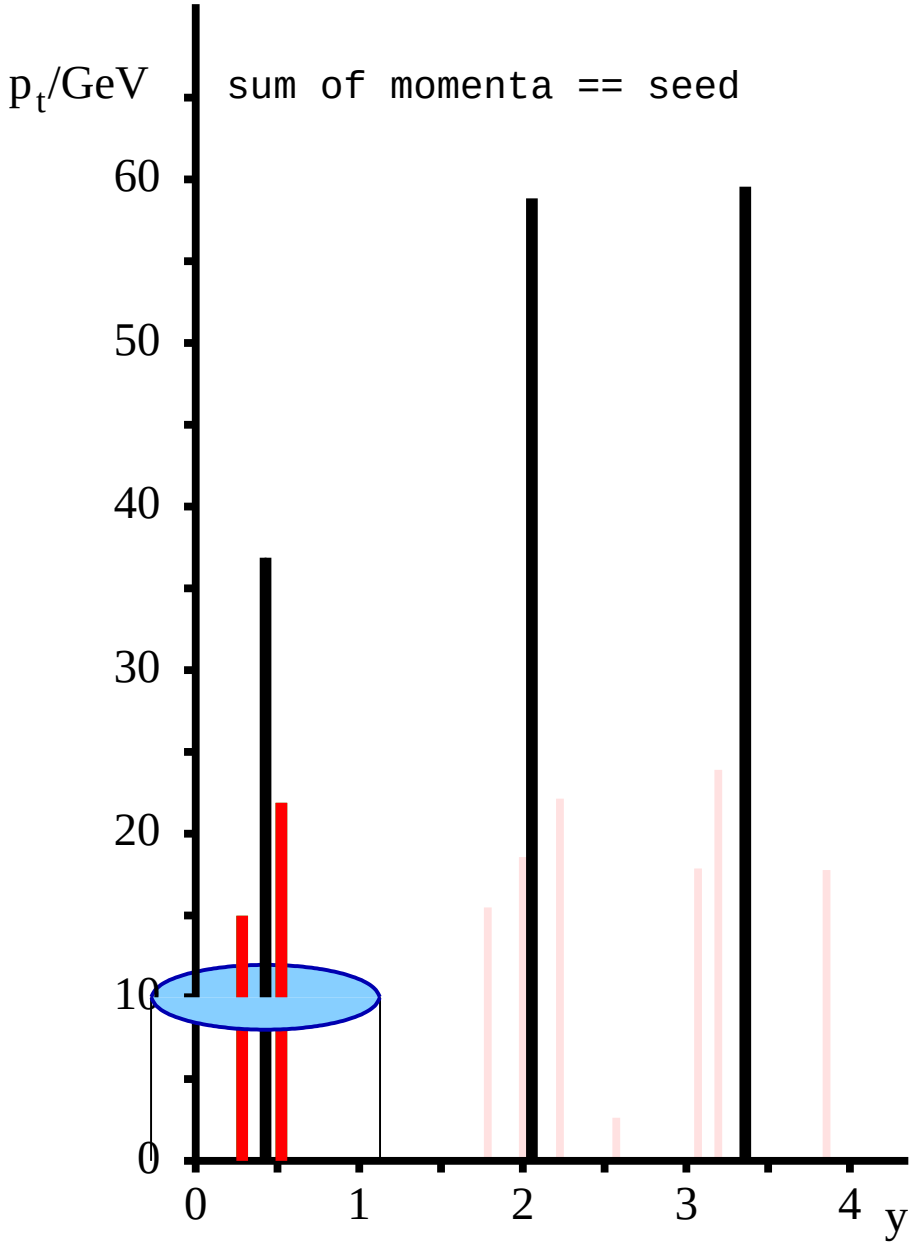
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe ... more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

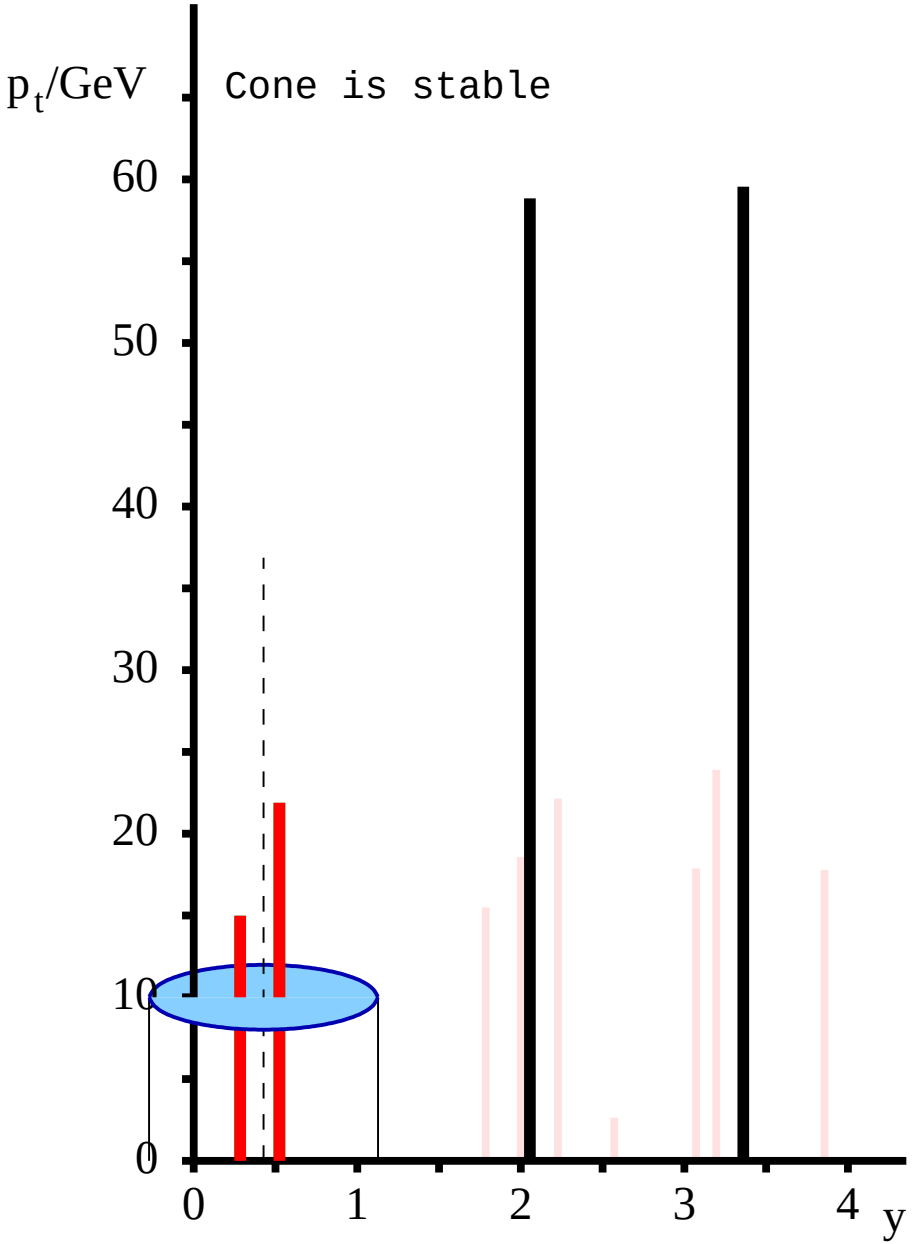
e.g. CMS iterative cone

- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

e.g. CMS iterative cone

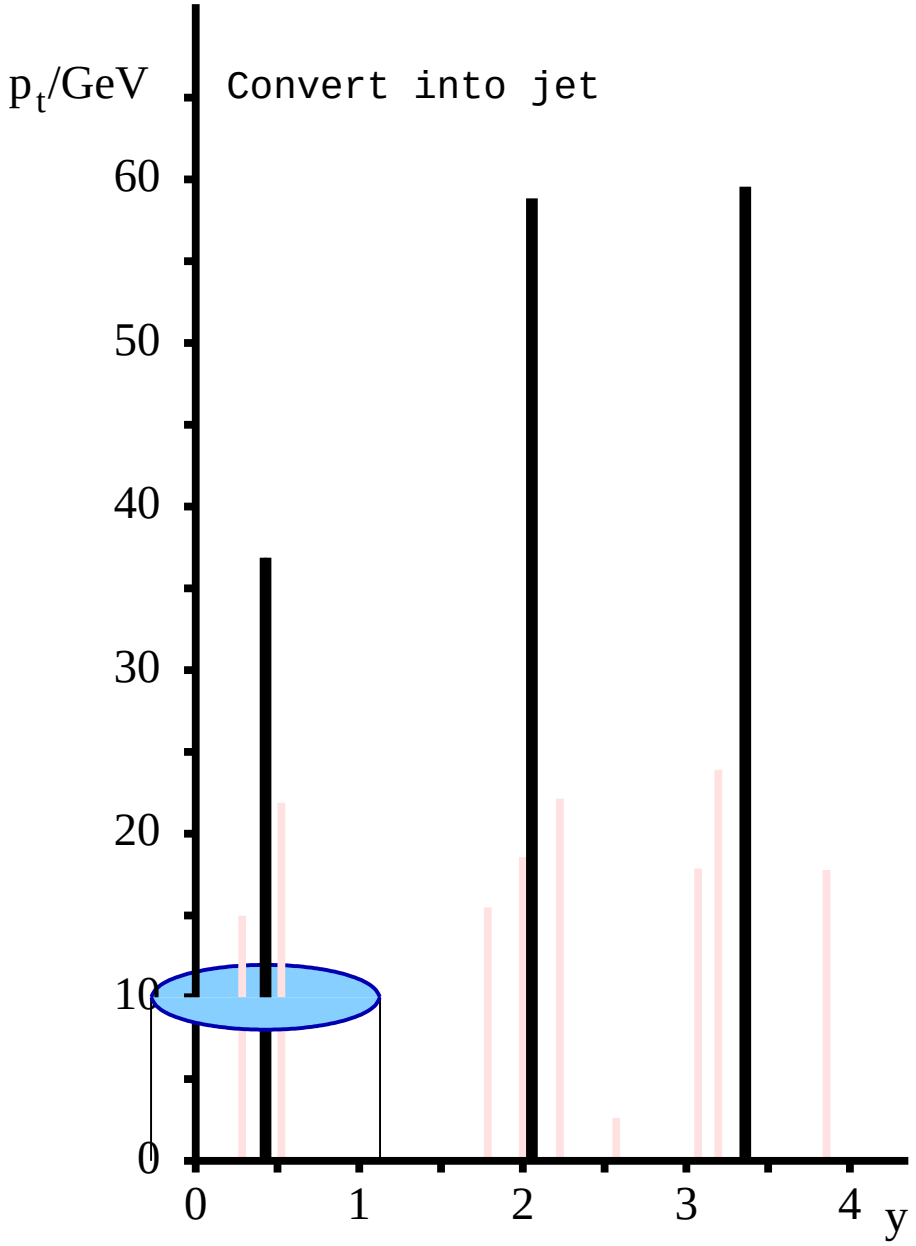
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe

more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

e.g. CMS iterative cone

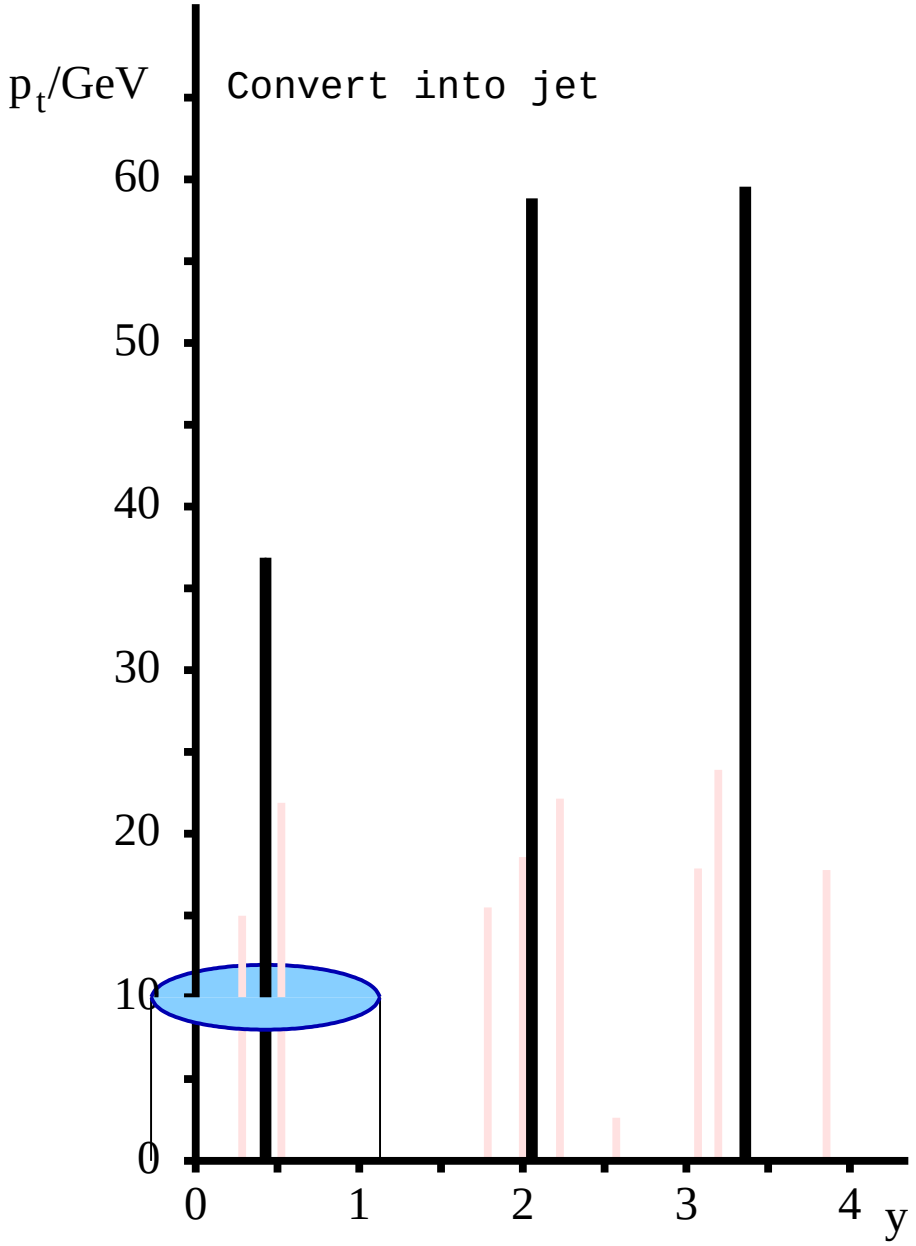
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe

more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

e.g. CMS iterative cone

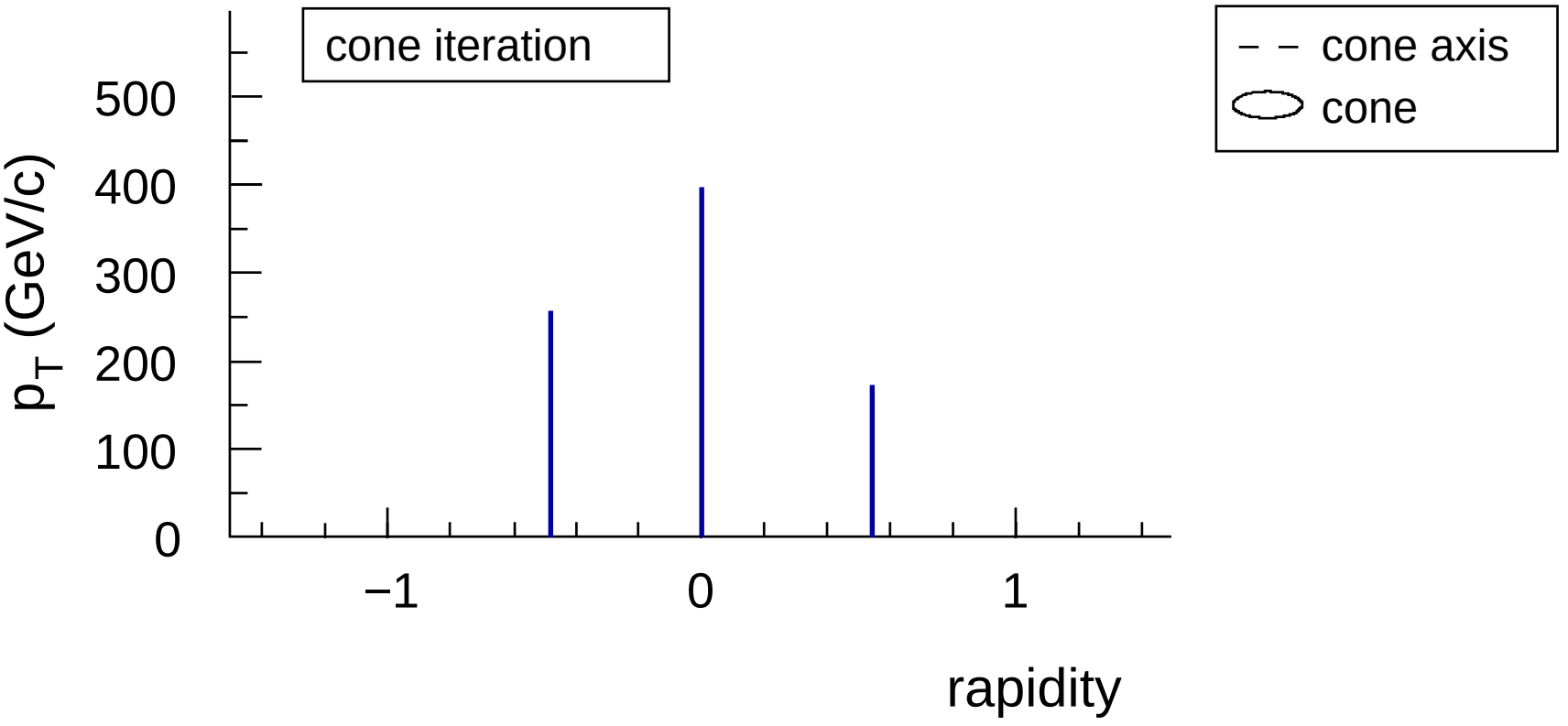
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe

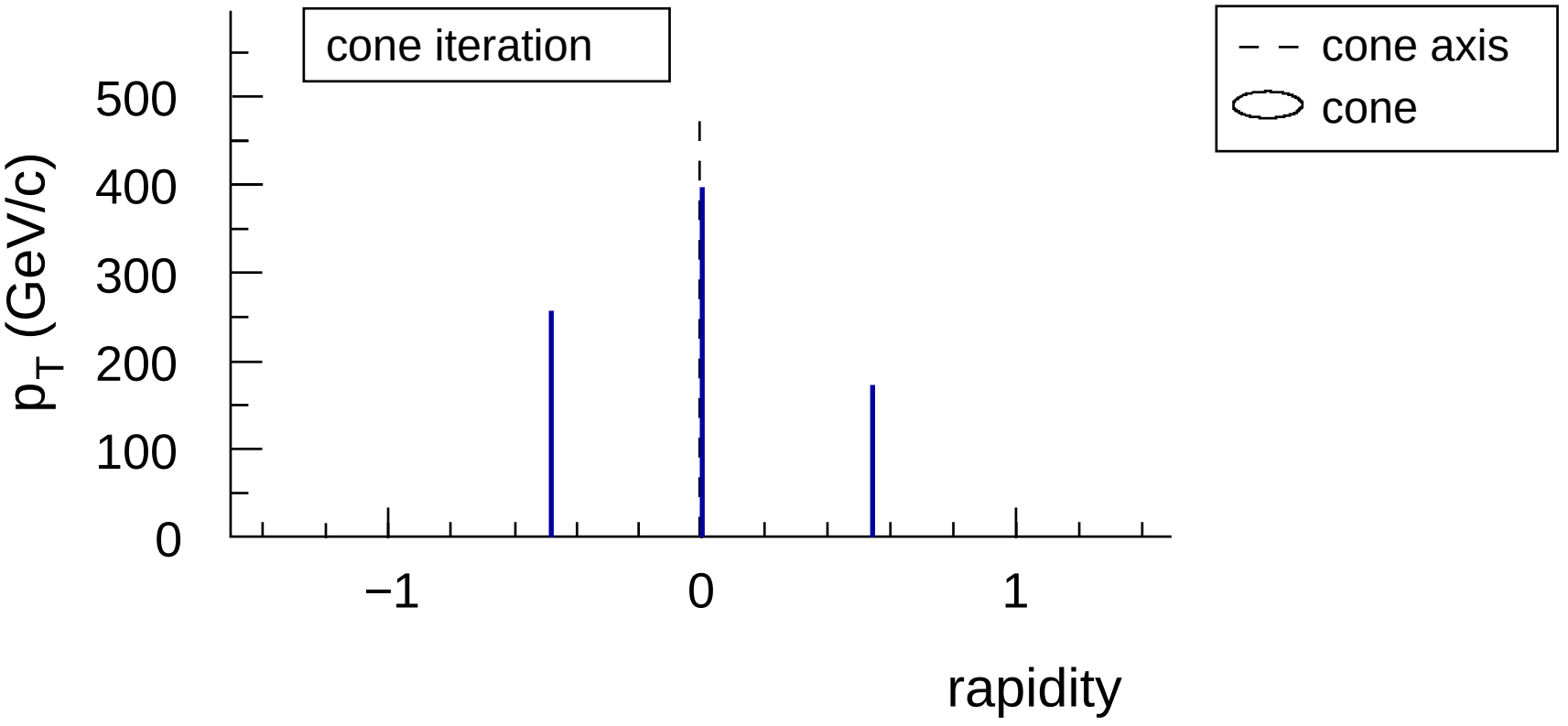
more right away...

ICPR iteration issue



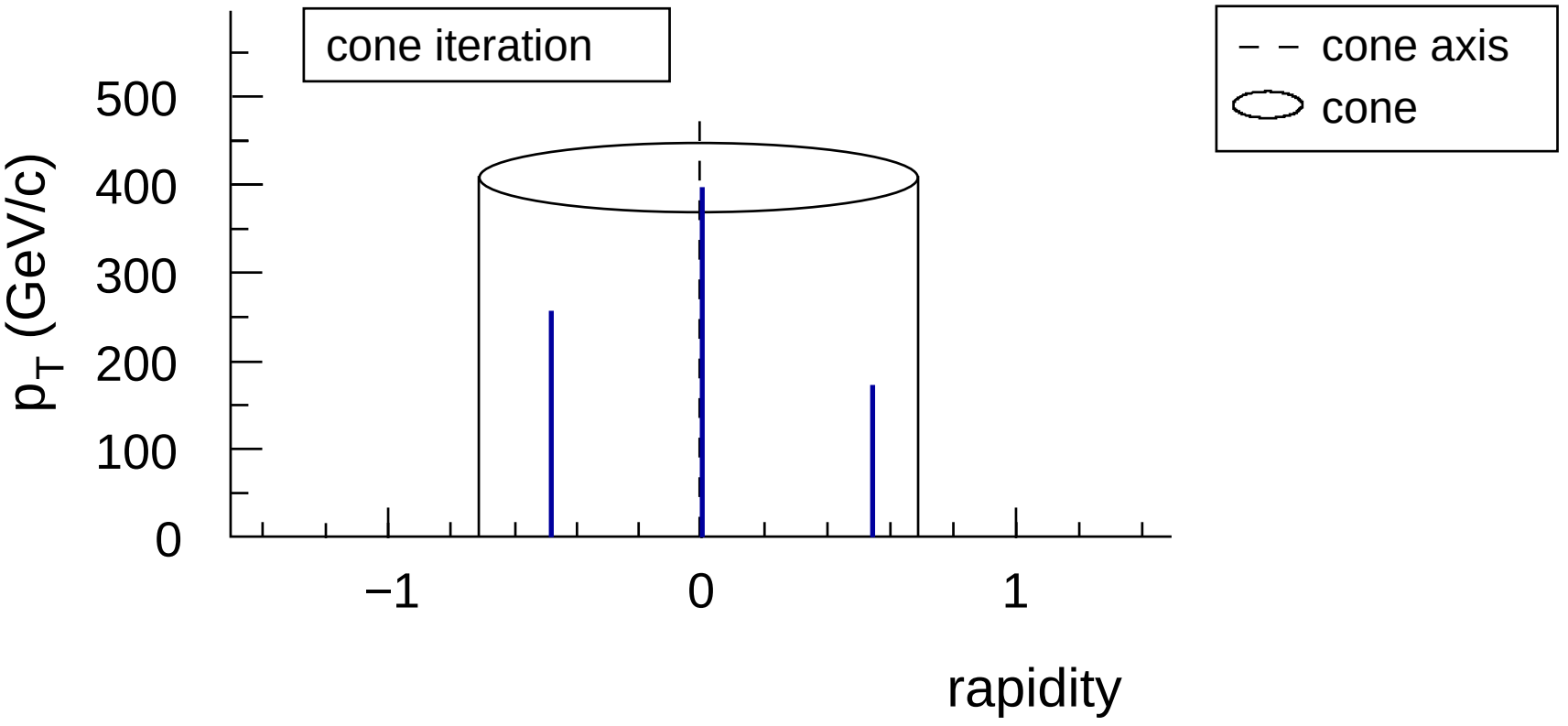
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



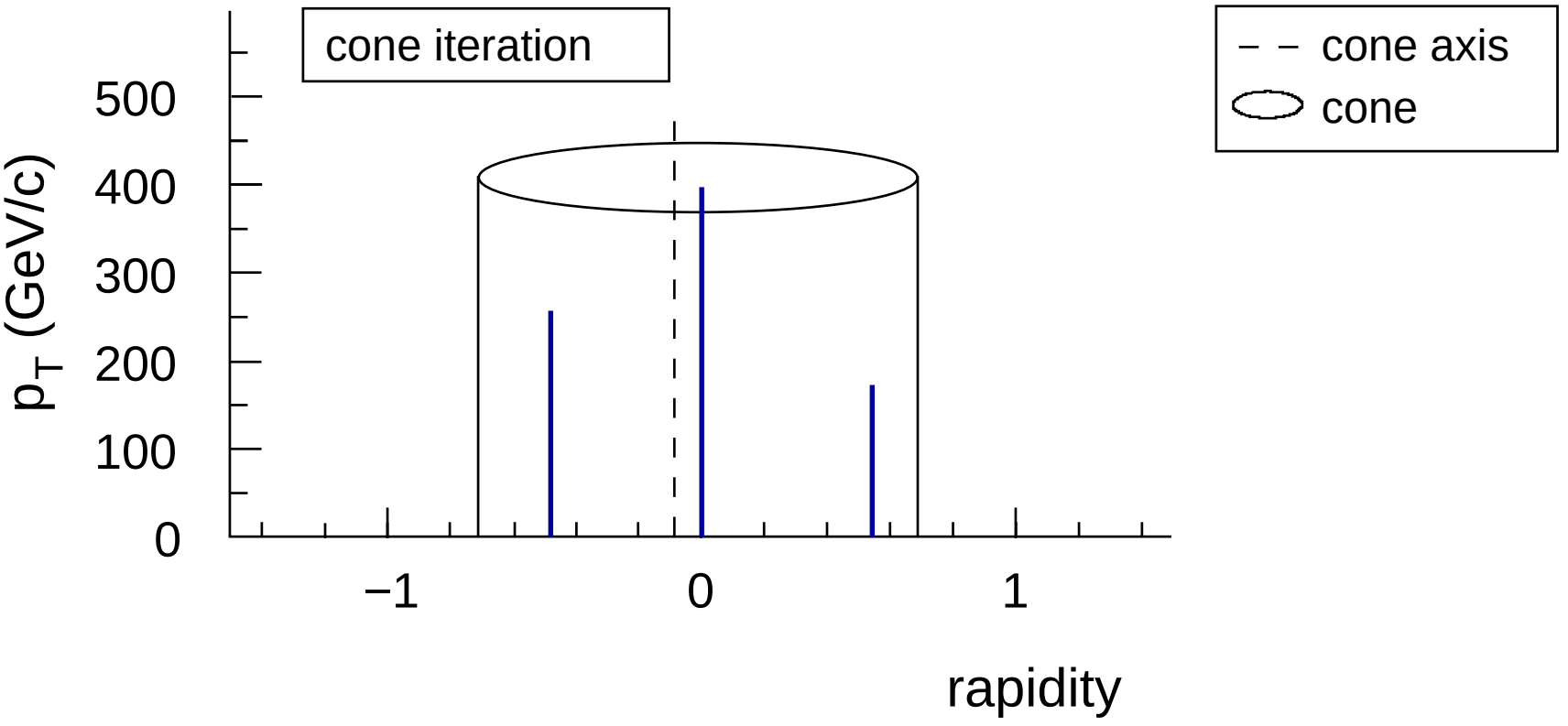
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



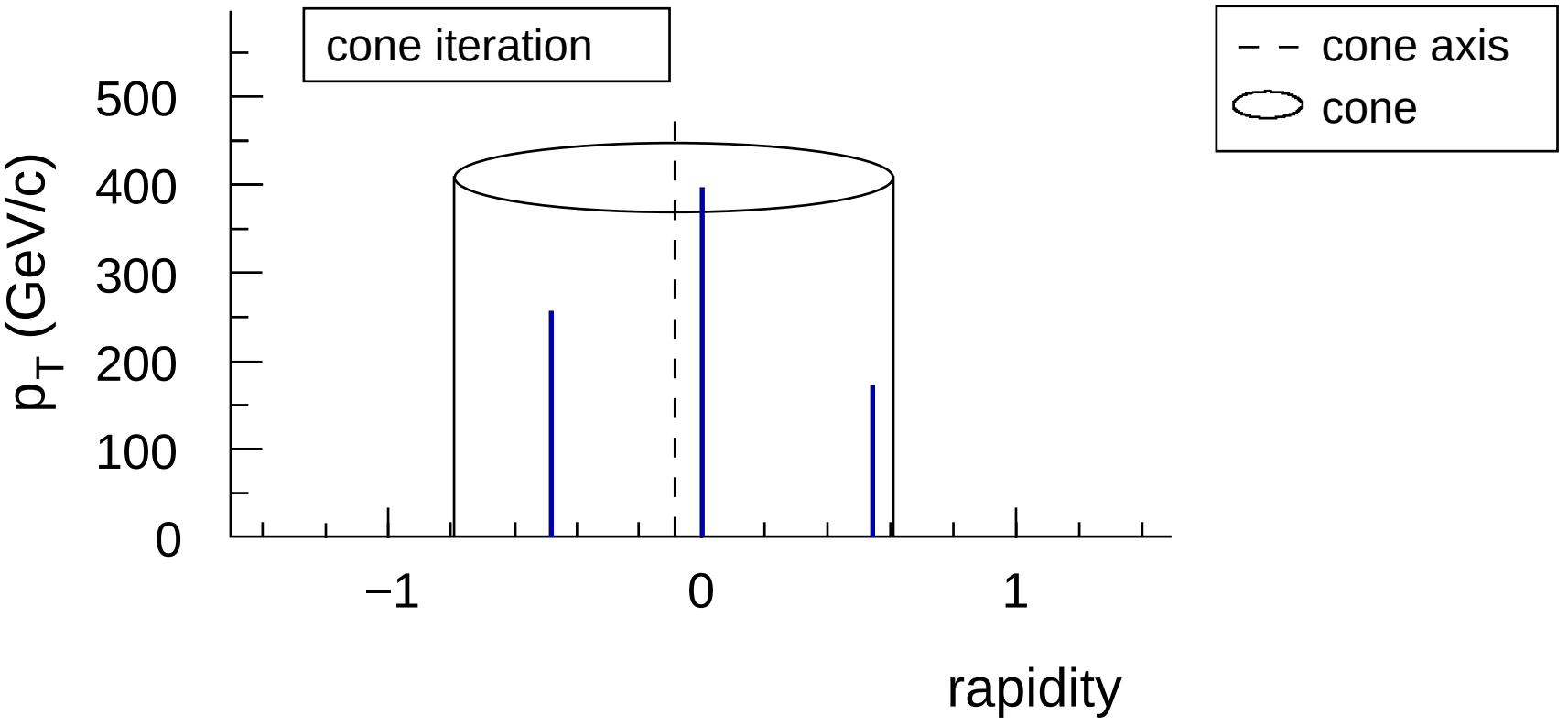
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



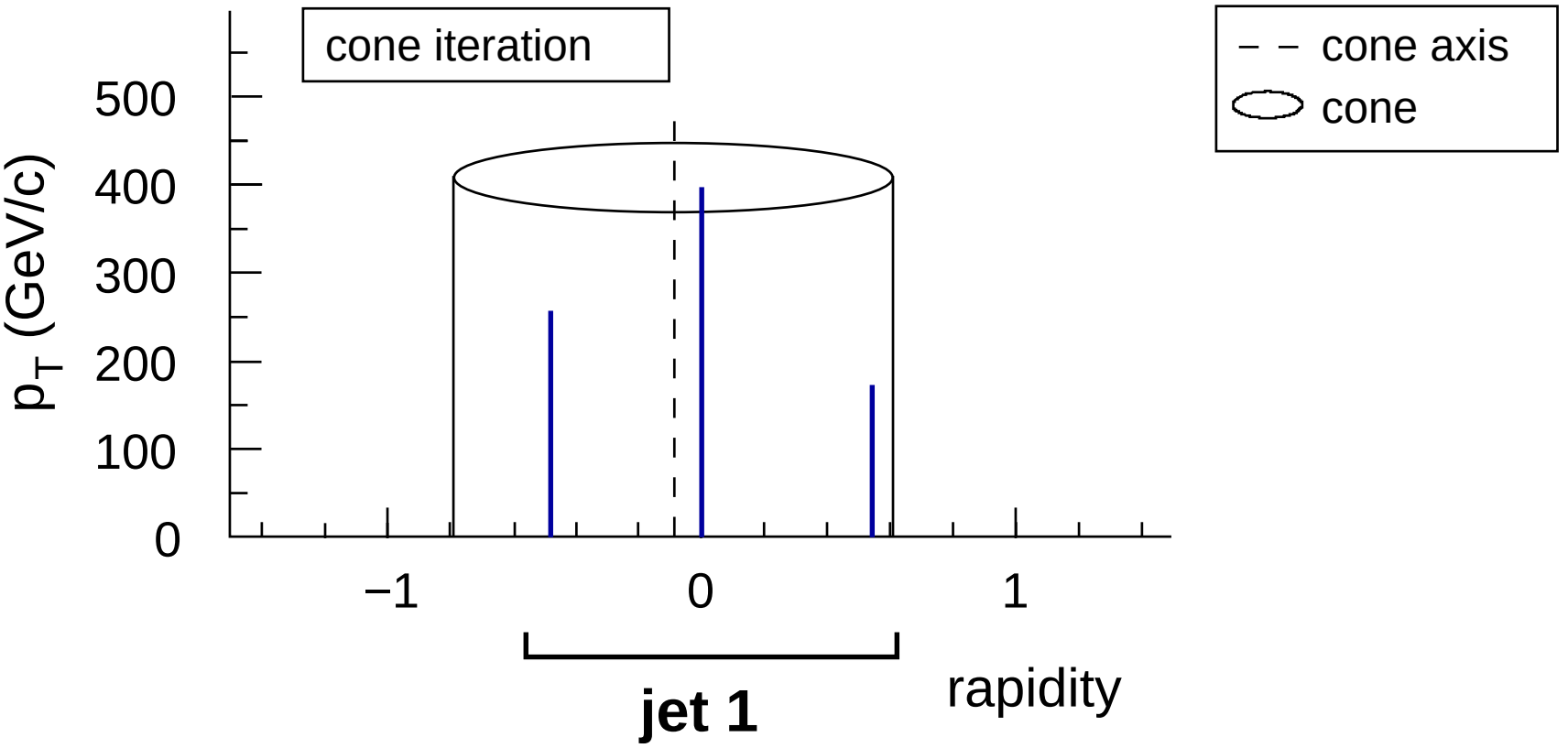
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



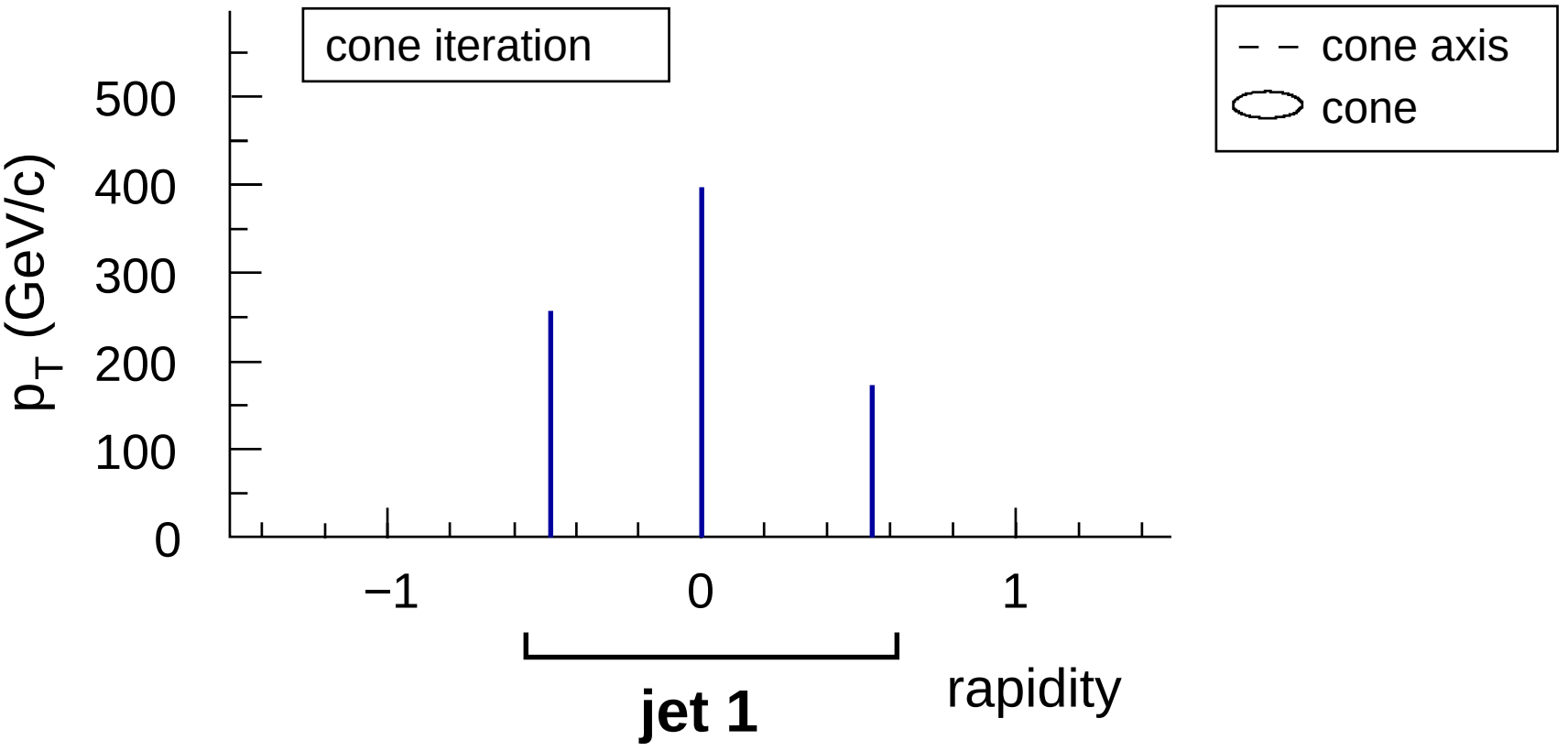
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



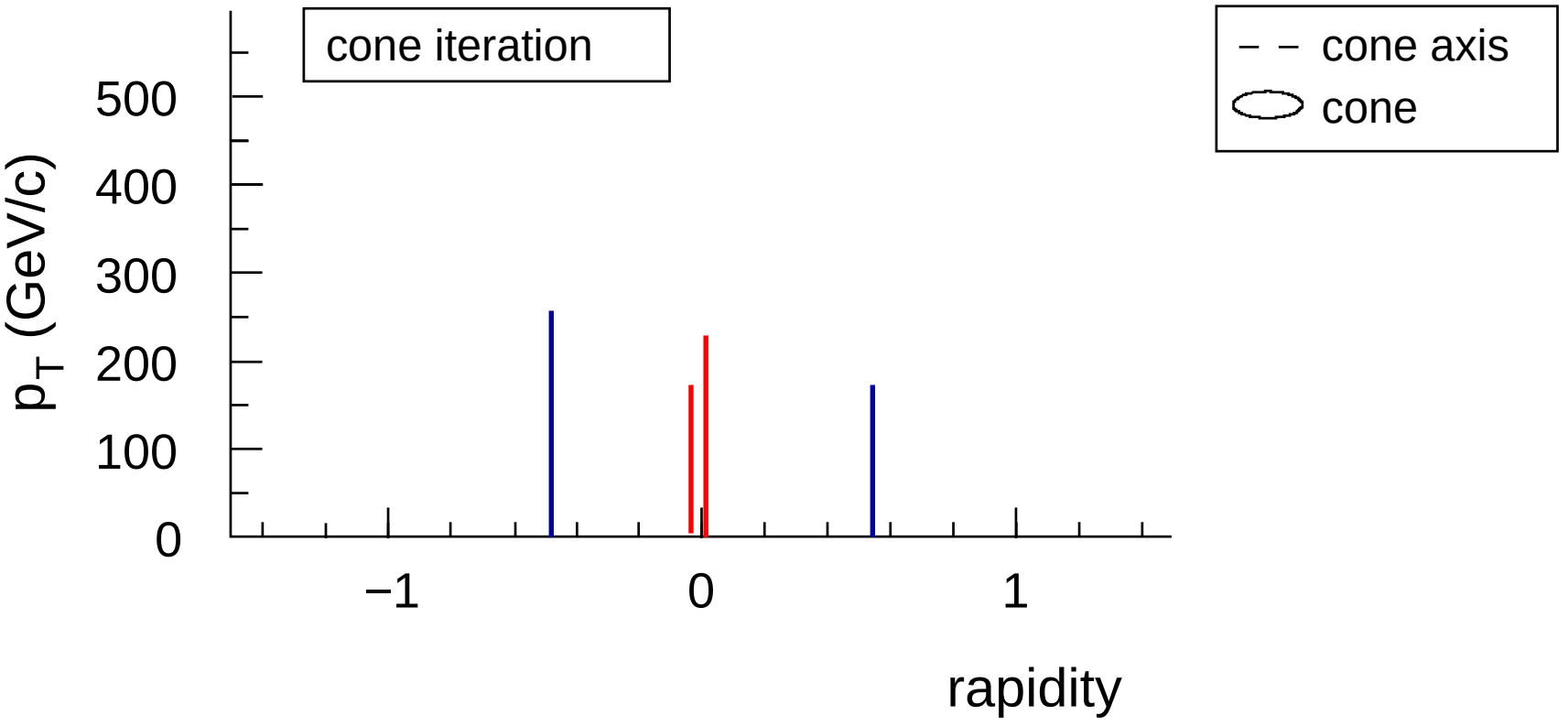
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



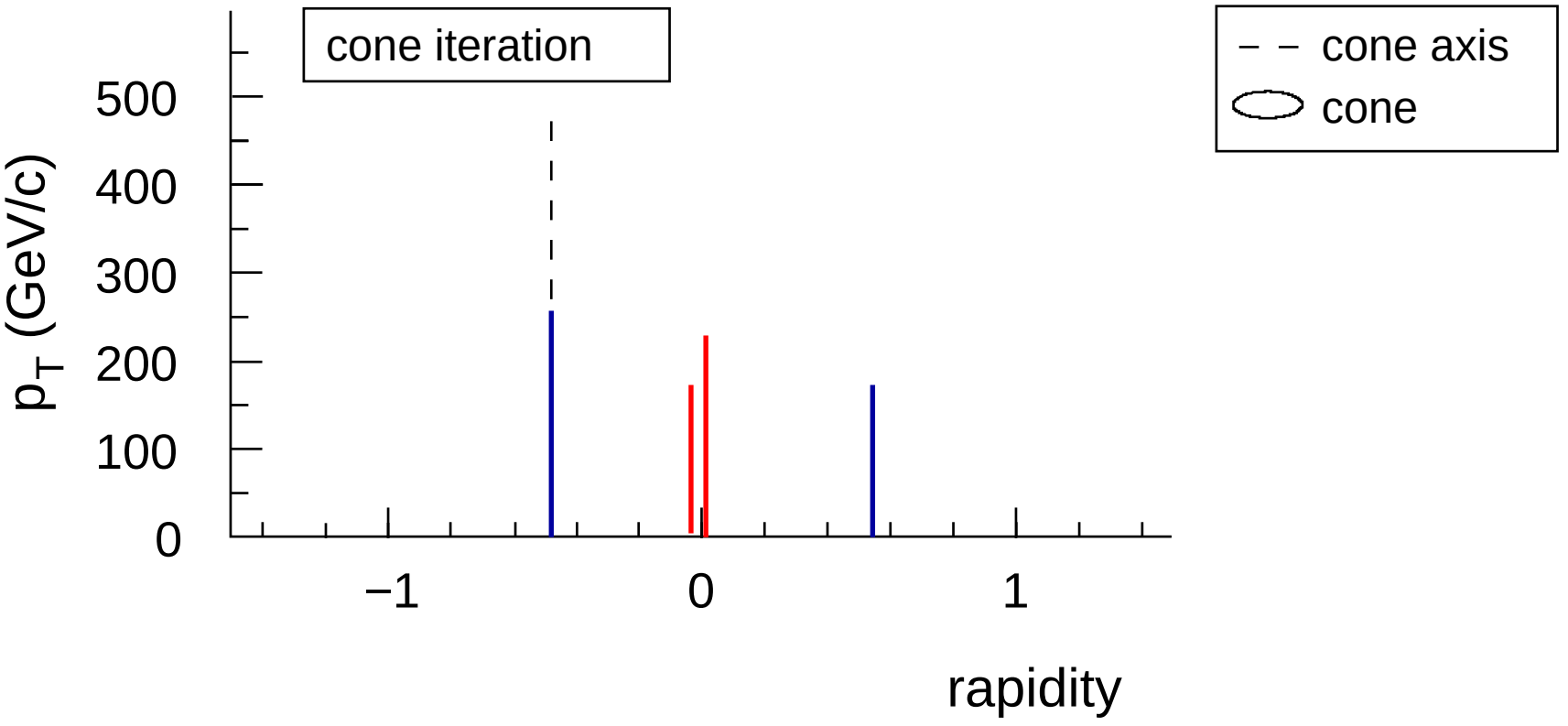
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



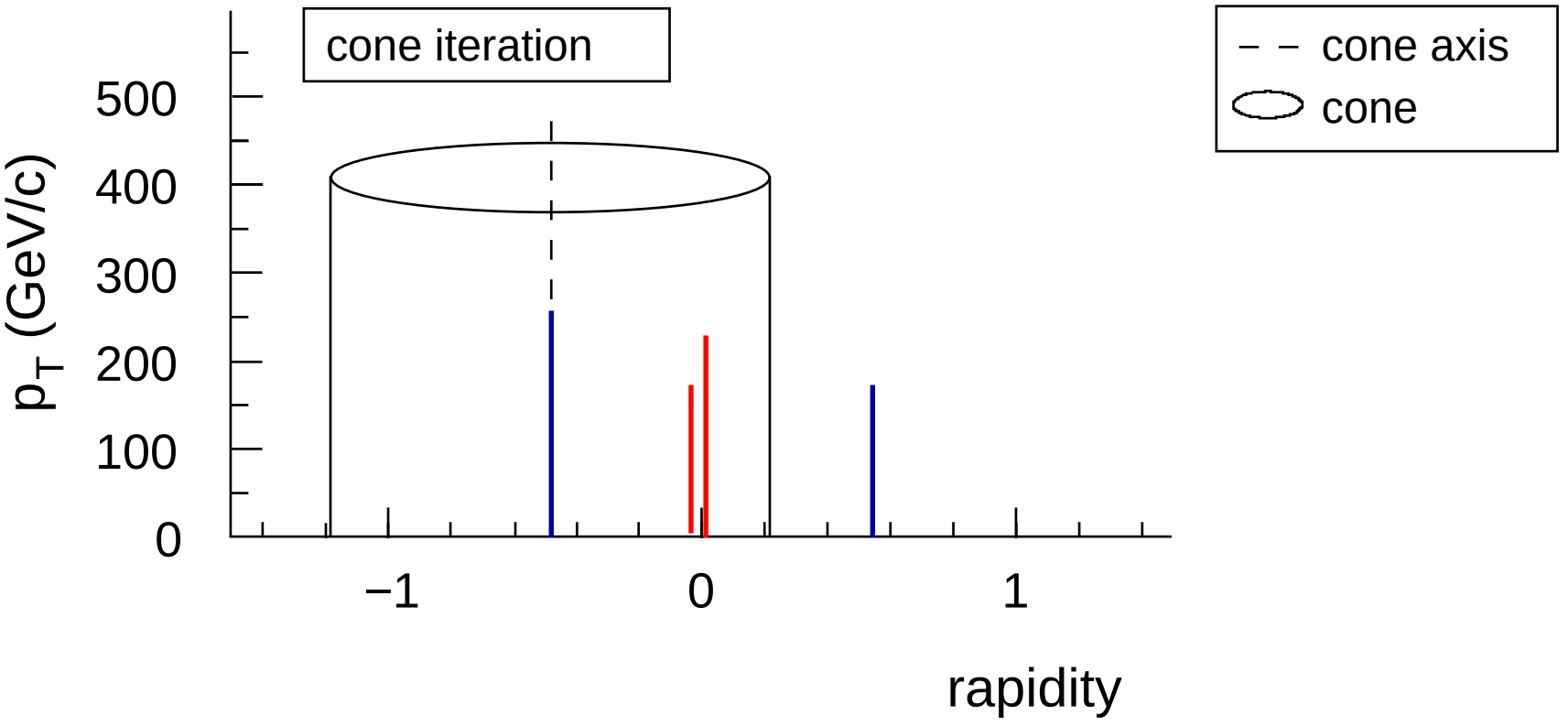
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



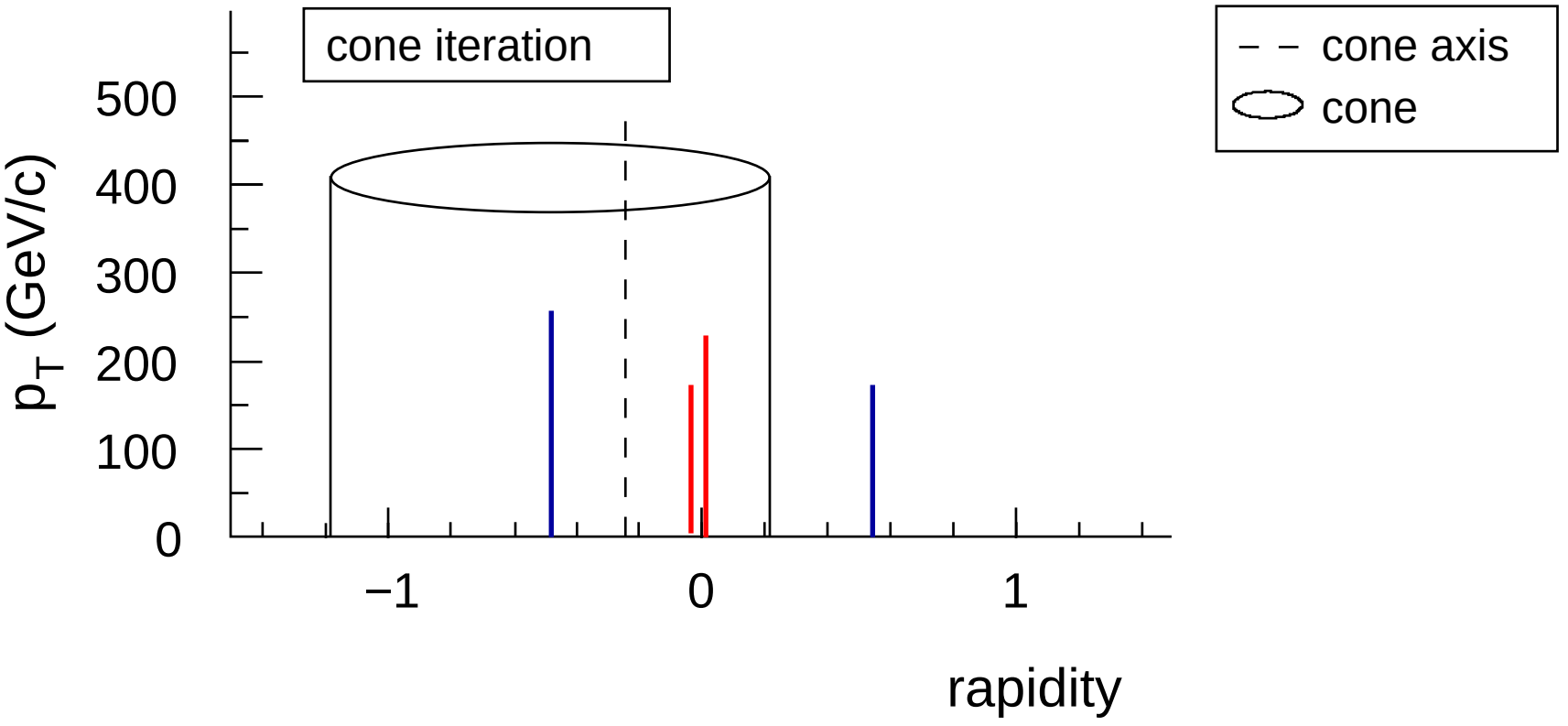
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



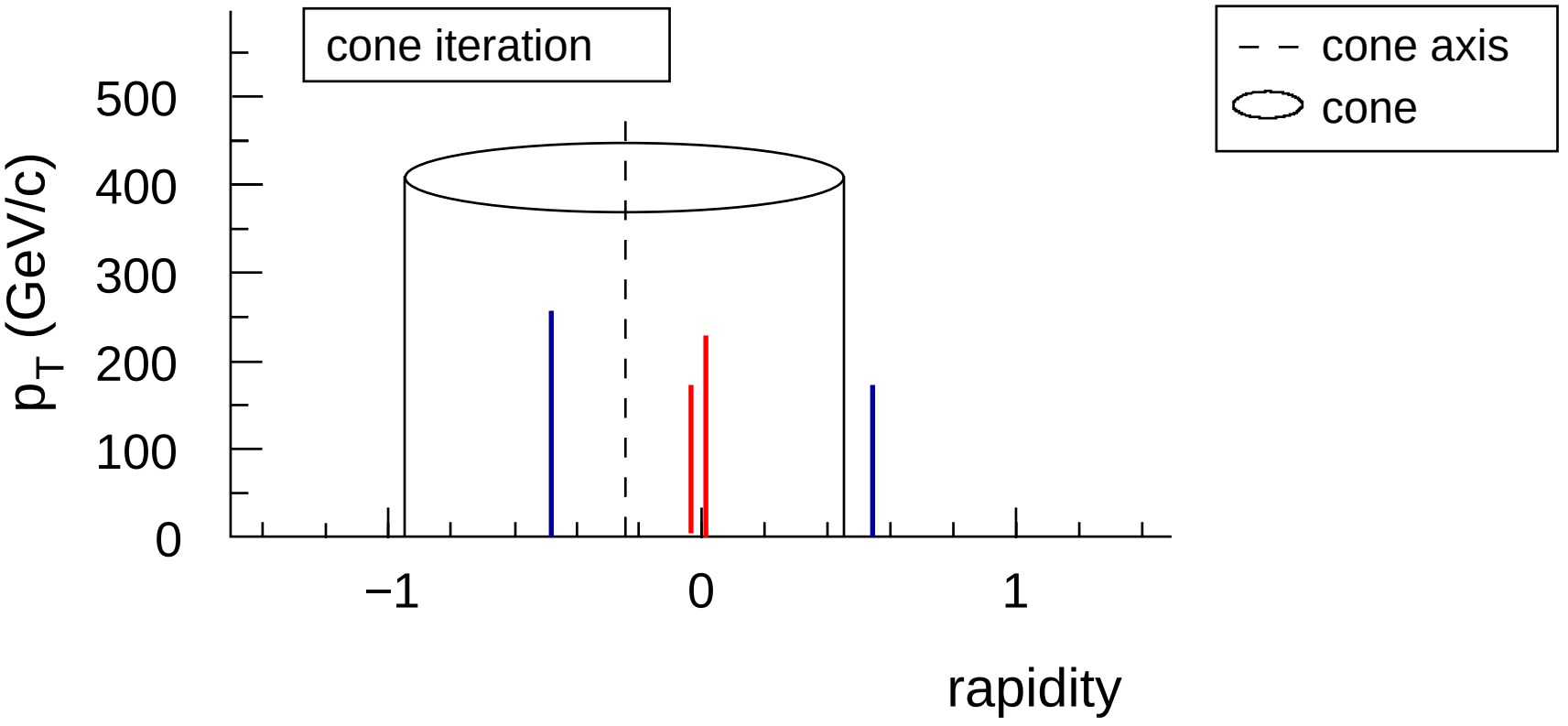
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



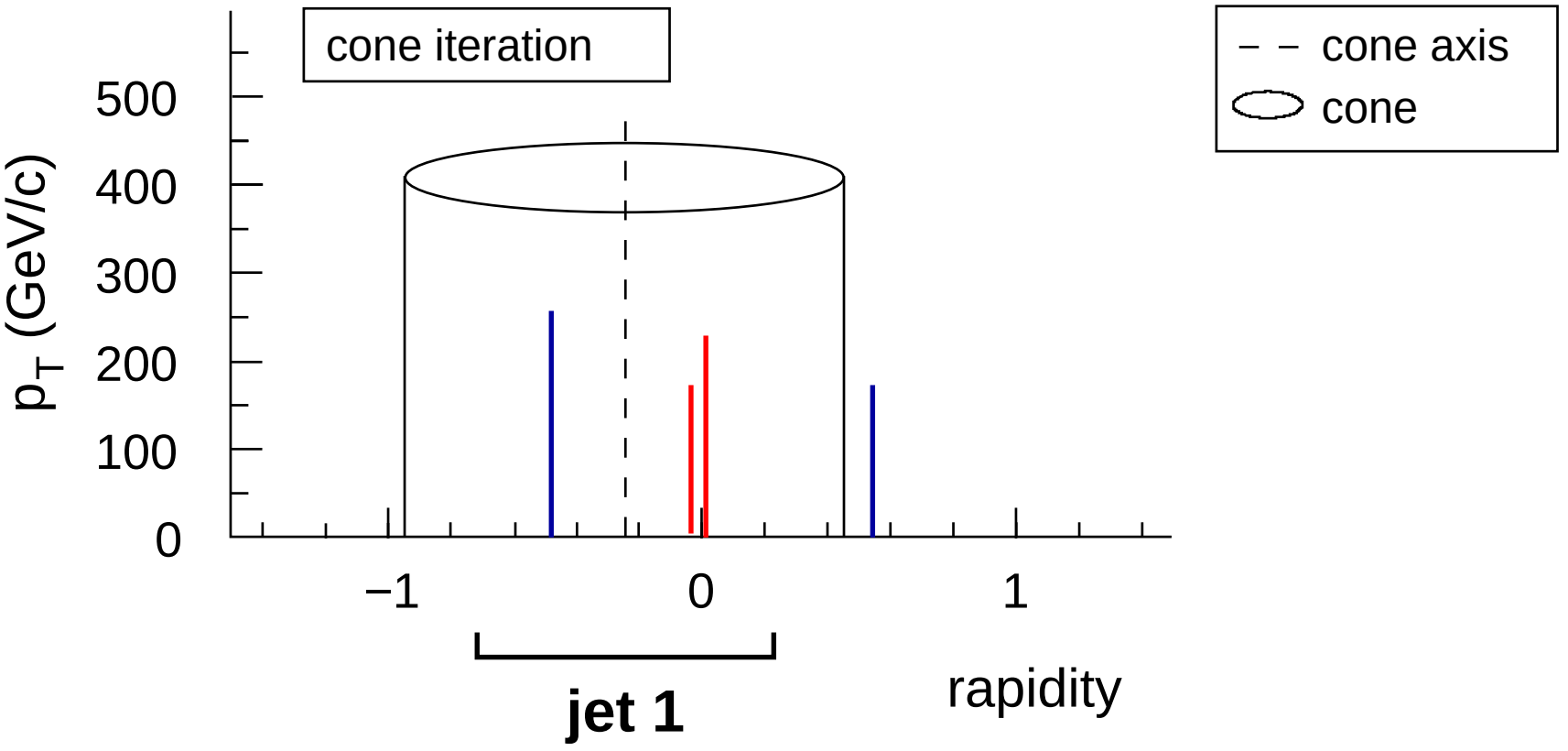
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



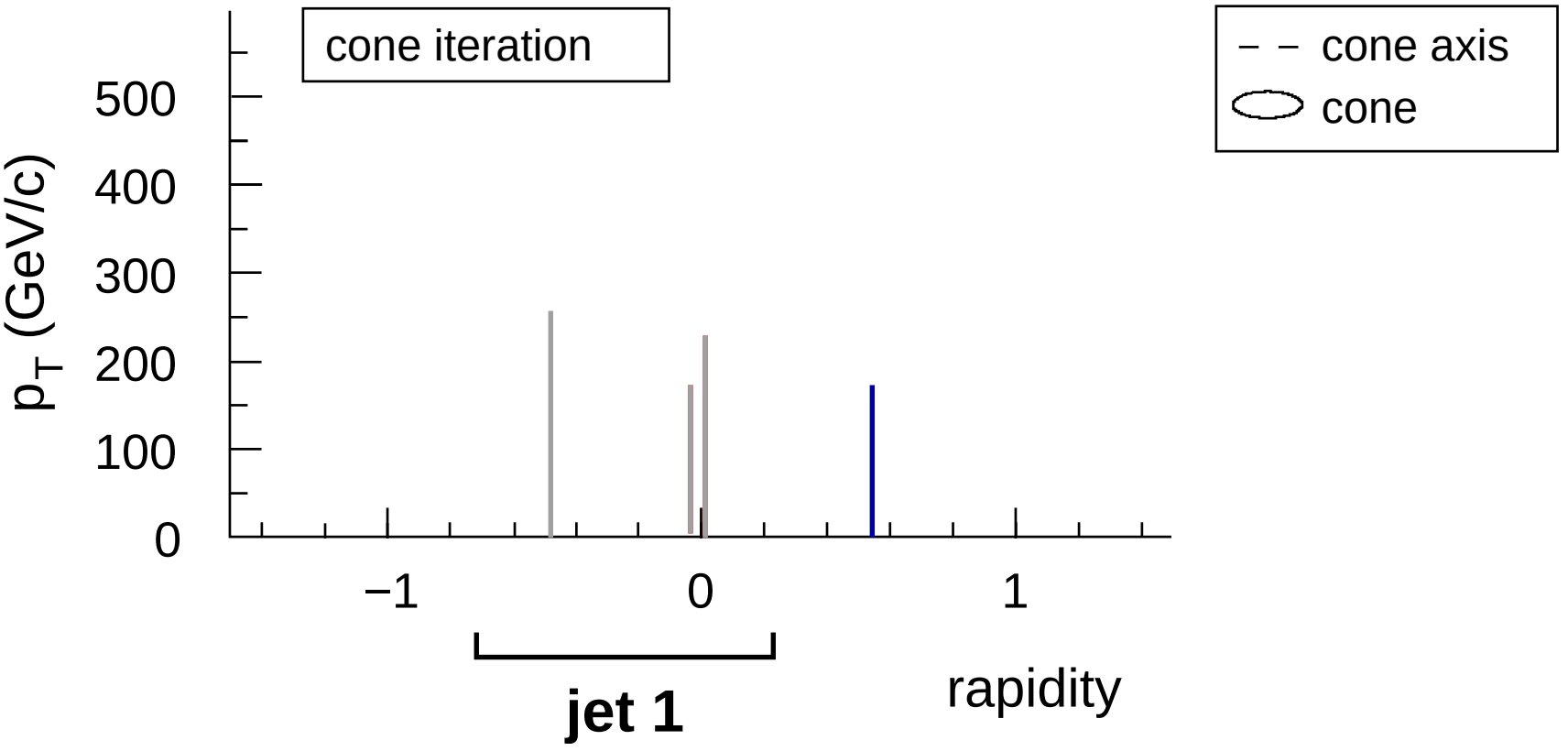
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



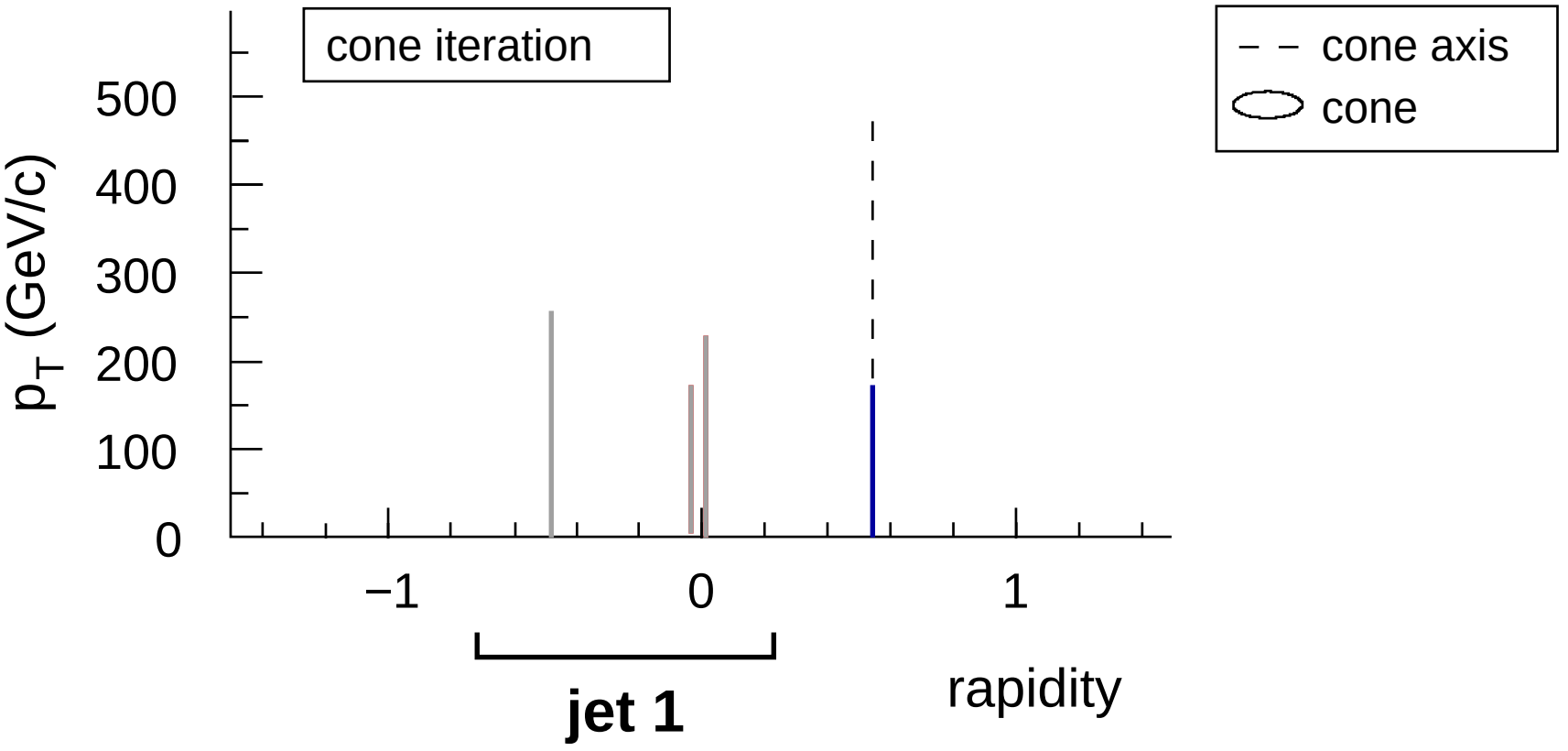
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



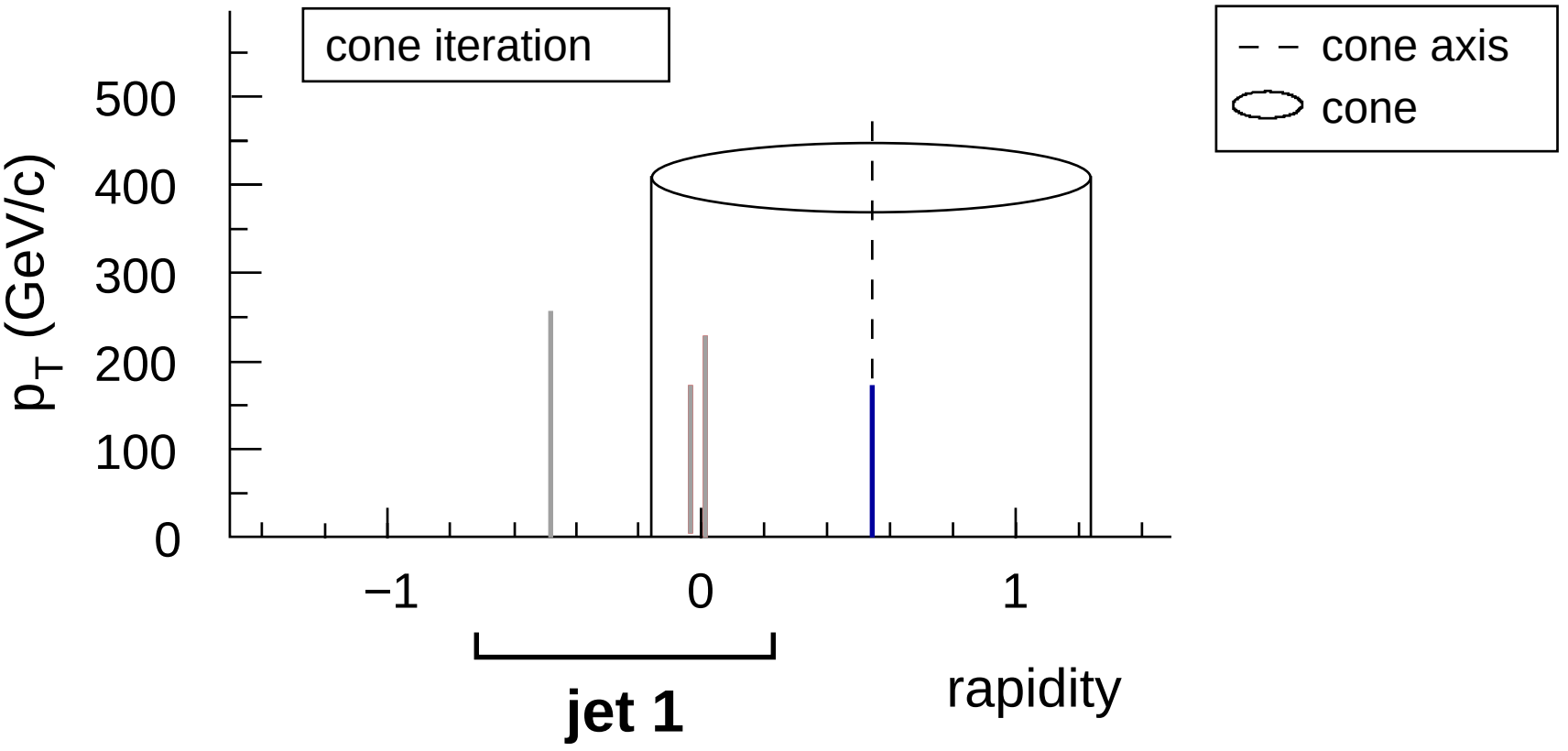
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



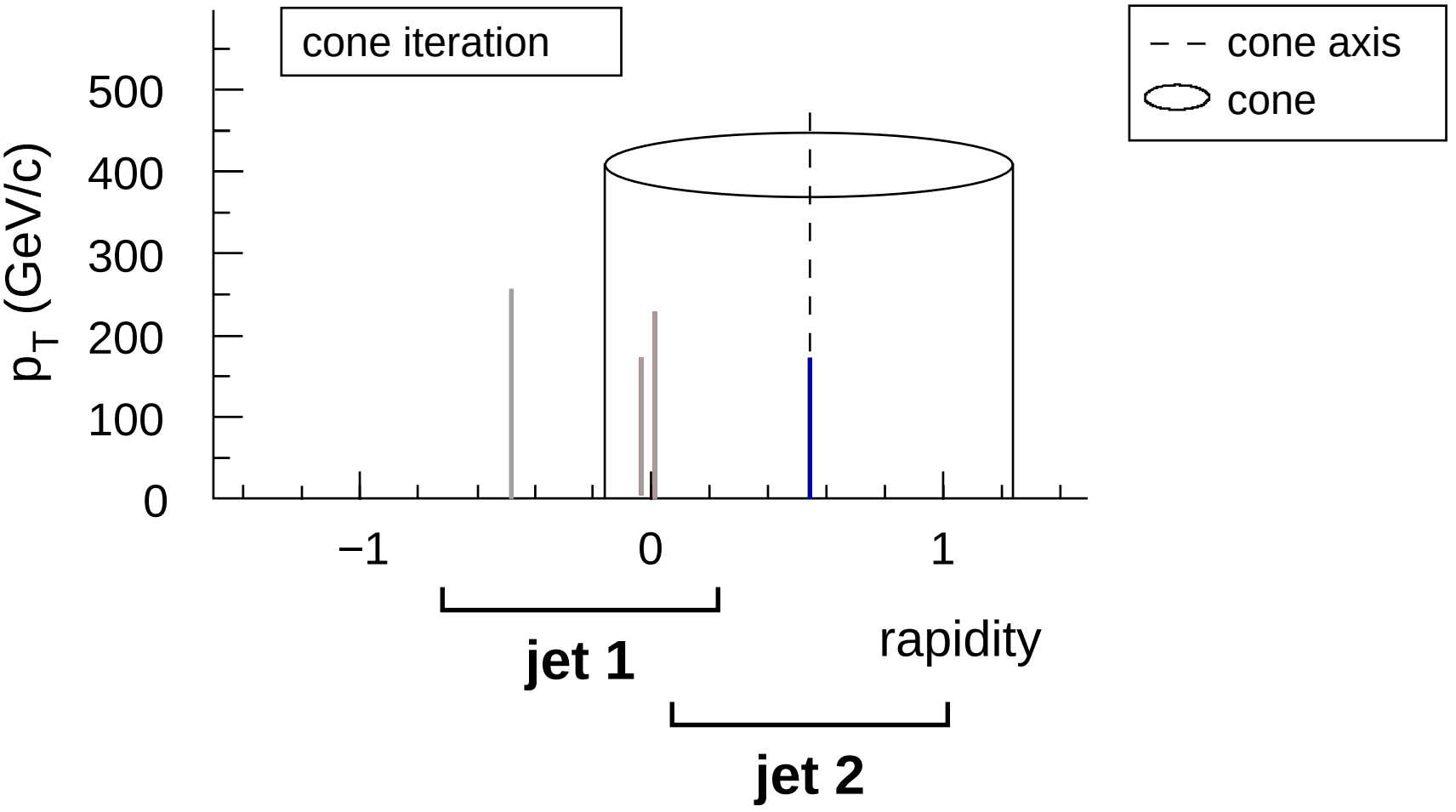
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



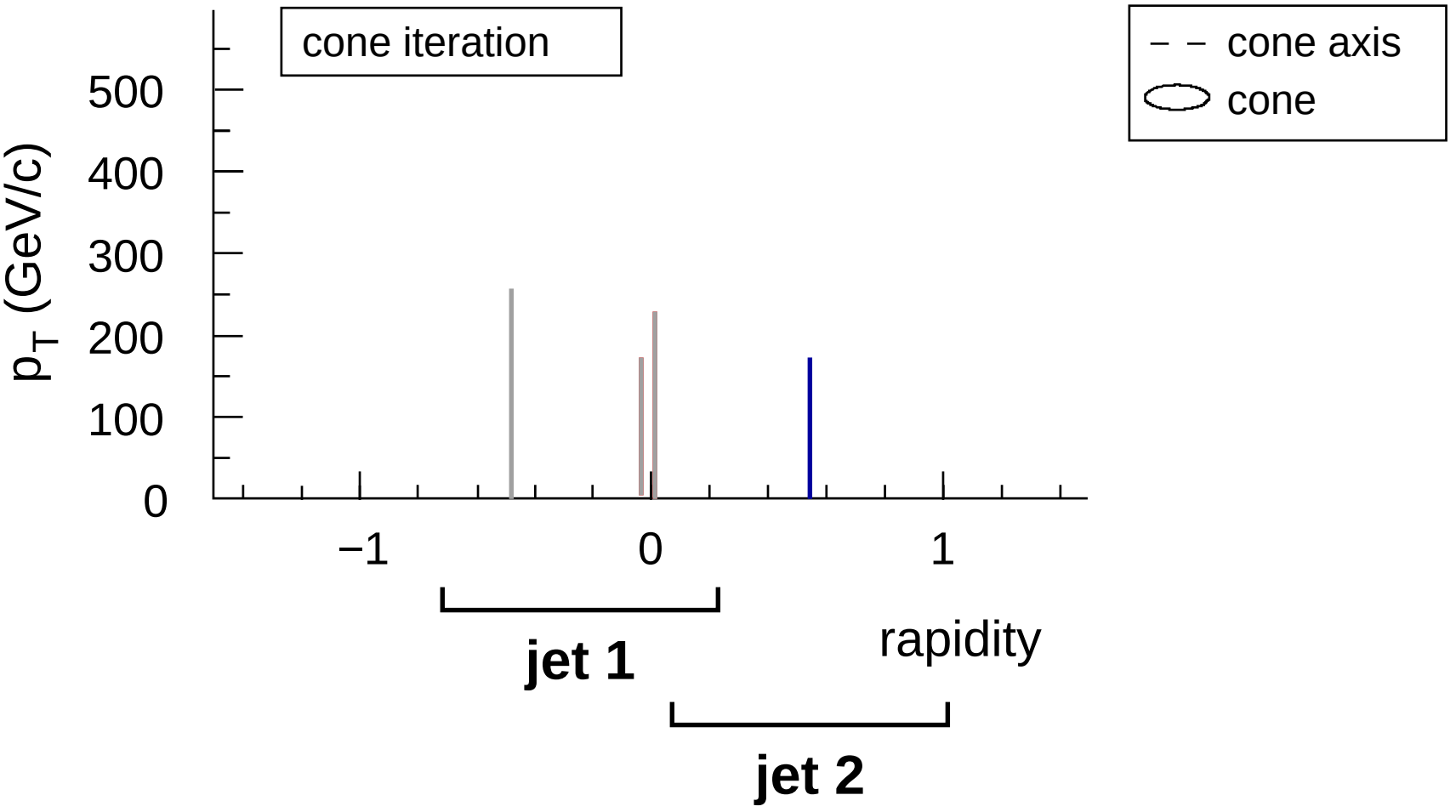
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



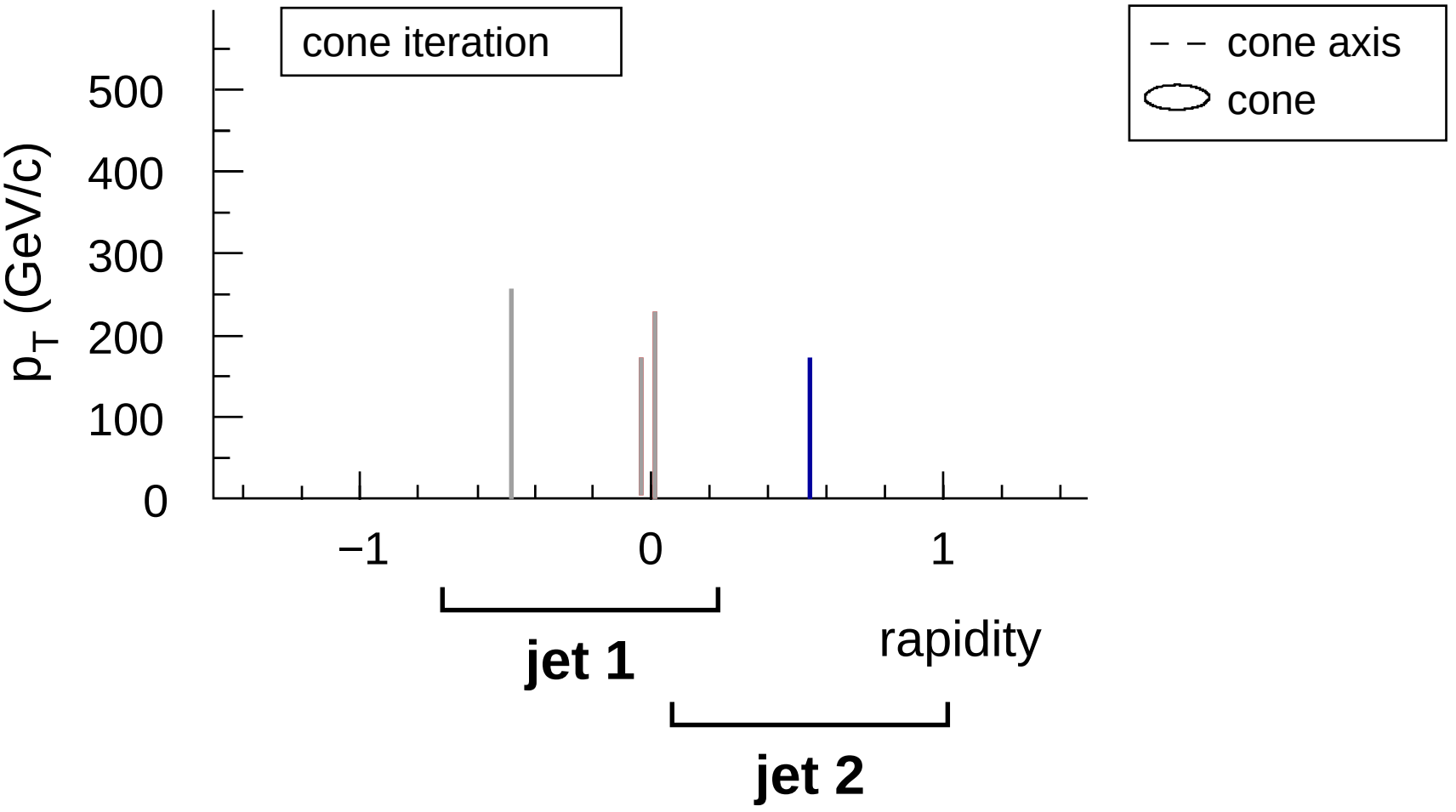
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

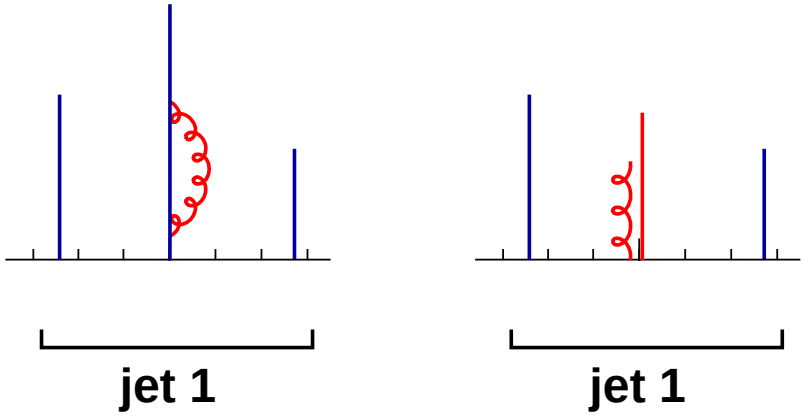
ICPR iteration issue



Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

Consequences of collinear unsafety

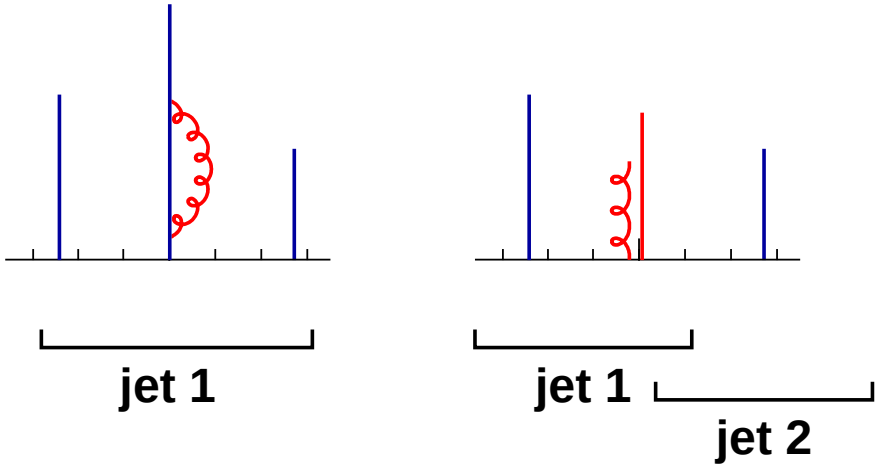
Collinear Safe



$\alpha_s^n \times (-\infty)$ $\alpha_s^n \times (+\infty)$

Infinities cancel

Collinear Unsafe



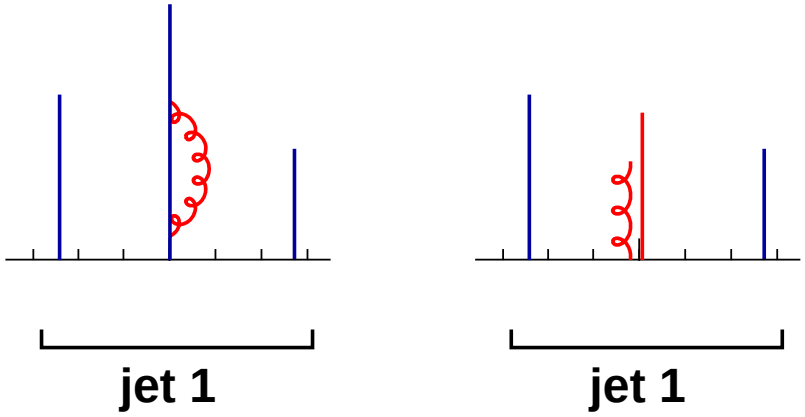
$\alpha_s^n \times (-\infty)$ $\alpha_s^n \times (+\infty)$

Infinities do not cancel

Invalidates perturbation theory

Consequences of collinear unsafety

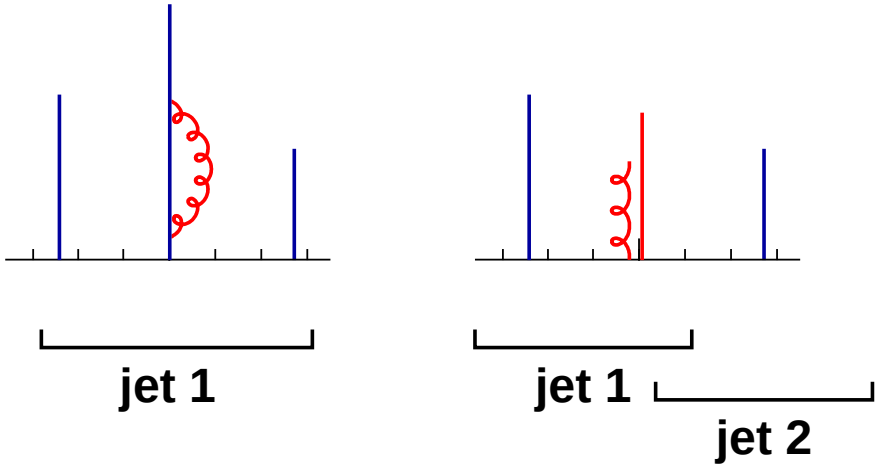
Collinear Safe



$\alpha_s^n \times (-\infty)$ $\alpha_s^n \times (+\infty)$

Infinities cancel

Collinear Unsafe



$\alpha_s^n \times (-\infty)$ $\alpha_s^n \times (+\infty)$

Infinities do not cancel

Invalidates perturbation theory

[End of lecture 1]

So far

- We've seen sequential recombination jet algorithms
- And we've started looking at cone algorithms and run into problems

Tomorrow

- Continue with the cones See more problems + some solutions
- Take a loot at the physics of jet algorithms