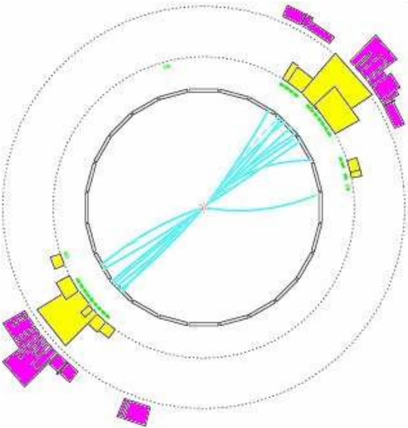
Jets

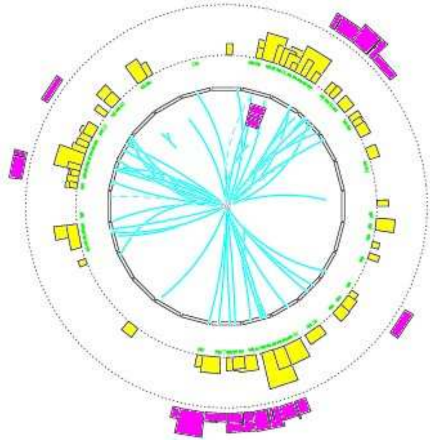
Gavin Salam

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

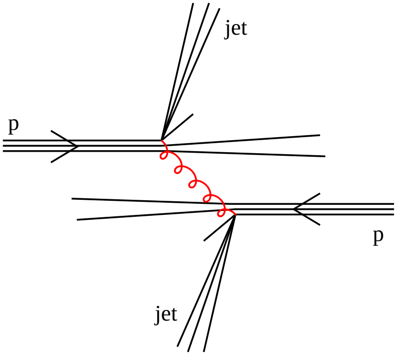
- precise process long way from being understood, even by lattice
- good models contain many parameters — complex process

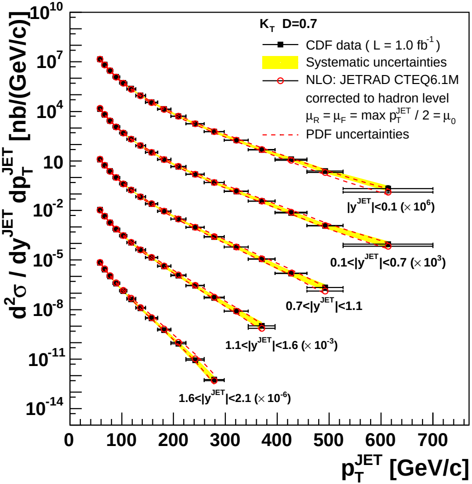
High-energy partons unavoidably lead to collimated bunches of hadrons.

See lectures by Dave Soper, Mike Seymour

Jets from scattering of partons

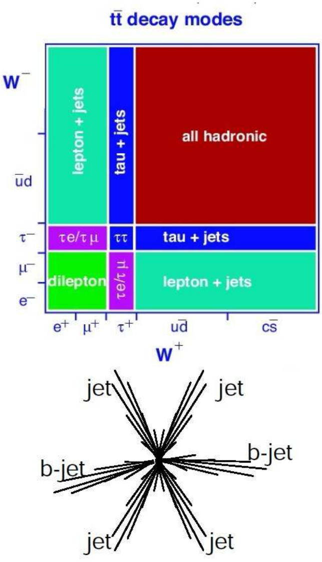
Jets are unavoidable at hadron colliders, e.g. from parton scattering





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

Jets from heavy decays



All-hadronic
(BR~46%, huge bckg)
picture: Juste LP05

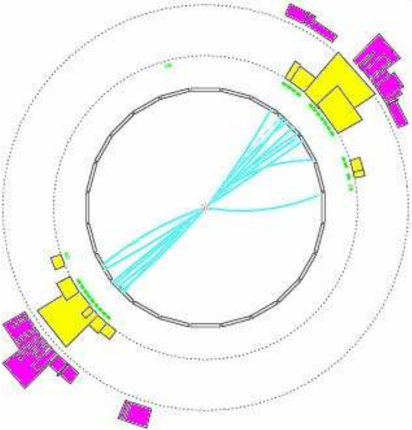
Heavy objects: multi-jet final-states

- 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

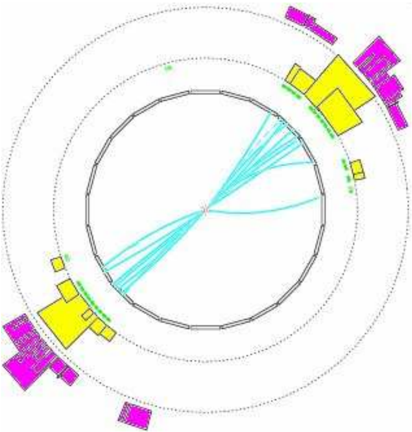
Seeing v. defining jets

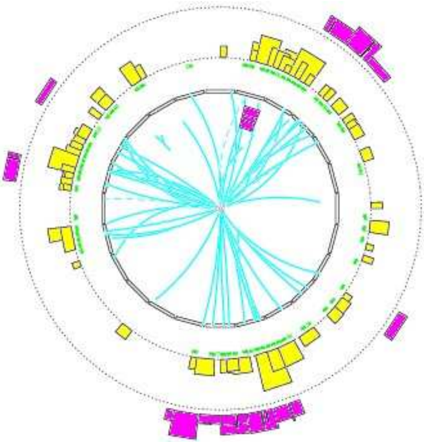


Jets are what we see.
Clearly(?) 2 jets here

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

Seeing v. defining jets

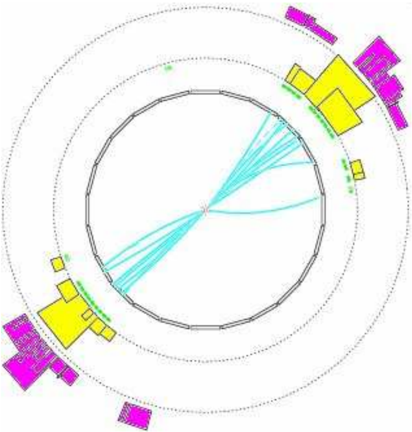




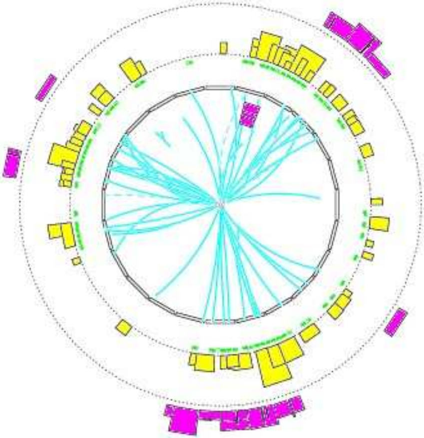
Jets are what we see.
Clearly(?) 2 jets here

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

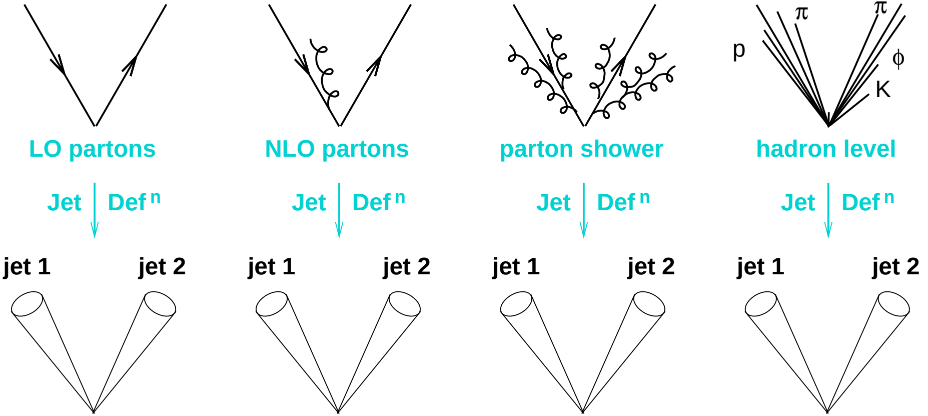


How many jets do you see?
Do you really want to ask yourself this question for 10^8 events?

Jet definitions

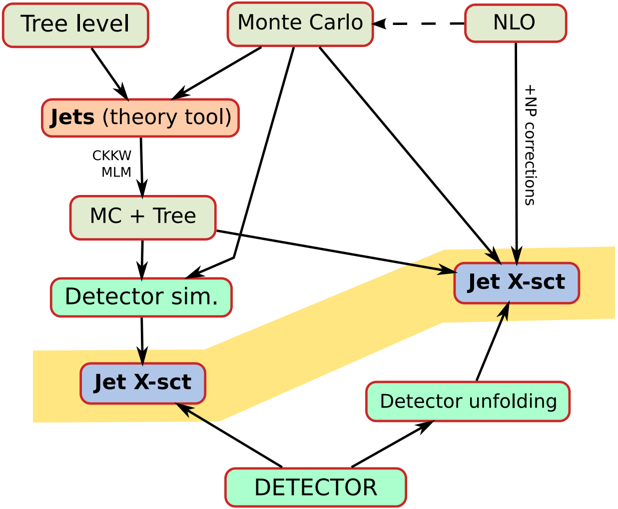
- *A jet definition is a fully specified set of rules for projecting information from 100's of hadrons, onto a handful of parton-like objects:*
  - or project 1000's of calorimeter towers
  - or project dozens of (showered) partons
  - or project a handful of (unshowered) partons
- Resulting objects (jets) used for many things, e.g. :
  - reconstructing decaying massive particles — e.g. top $\rightarrow$ 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



Projection to jets should be resilient to QCD effects

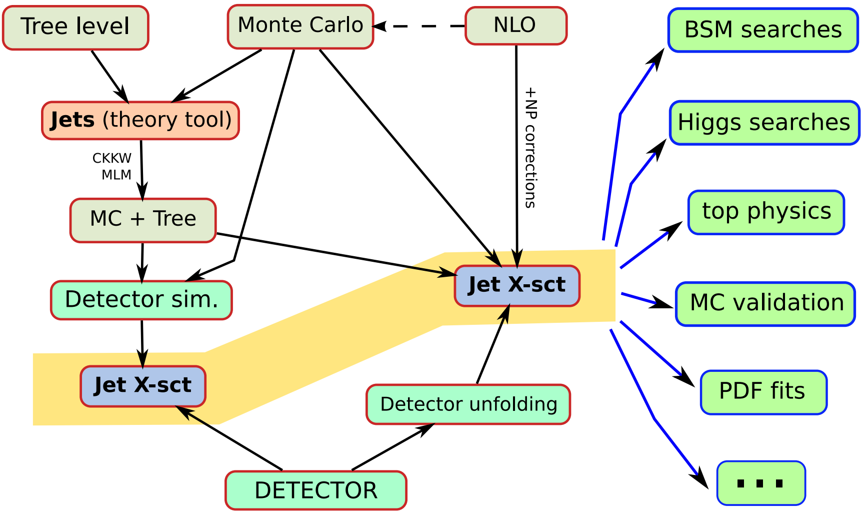
QCD jets flowchart



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

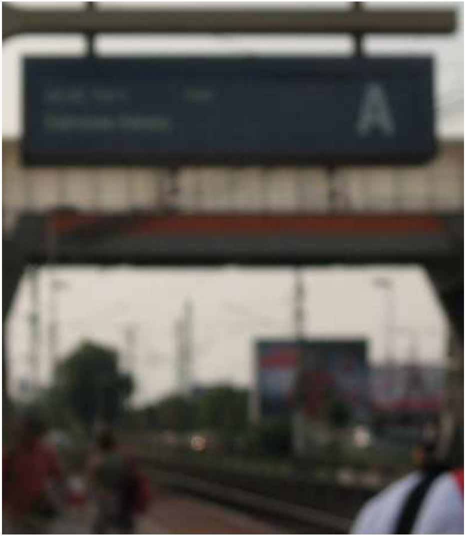
- Fine detail on bus ticket to train station — shoot from close up, focus = 40cm

[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 → 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 $\rightarrow$ 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

<u>Why?</u>

- Because otherwise lose real-virtual cancellation in NLO/NNLO QCD calculations $\rightarrow$ 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 **$\mathbf{k_t}$/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

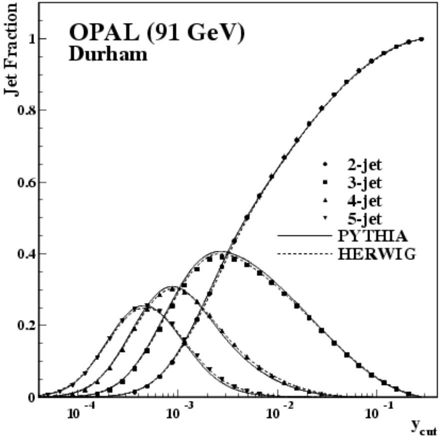
Catani, Dokshitzer, Olsson, Turnock & Webber '91

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

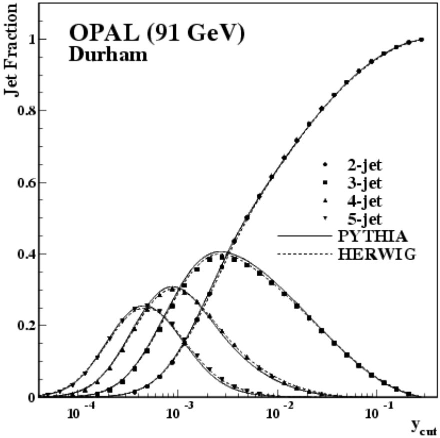


Most widely-used jet algorithm in e^+e^-

- 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 $d_i j$ 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

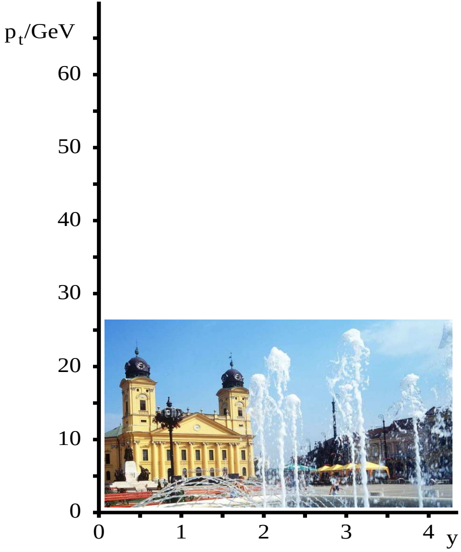
$\mathbf{k_t}$ alg.: Find smallest of

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

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



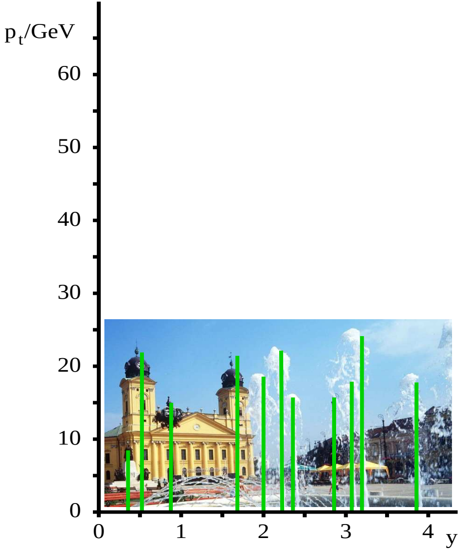
k_t alg.: Find smallest of

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

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



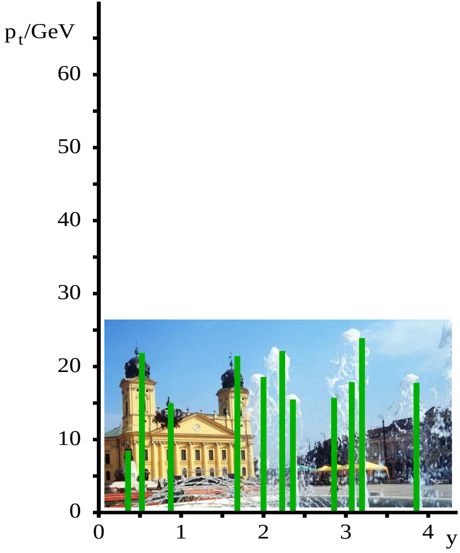
k_t alg.: Find smallest of

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

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



$\mathbf{k_t}$ alg.: Find smallest of

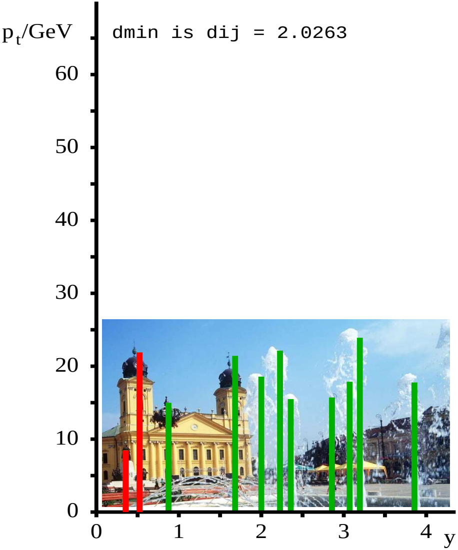
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



$\mathbf{k_t}$ alg.: Find smallest of

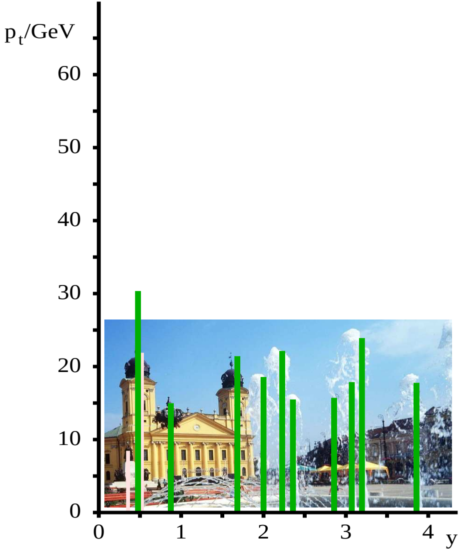
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

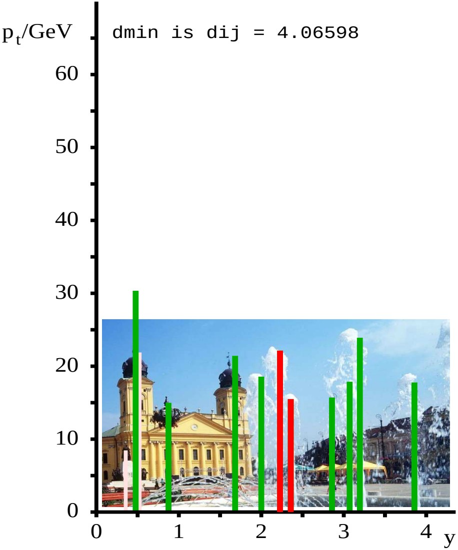
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

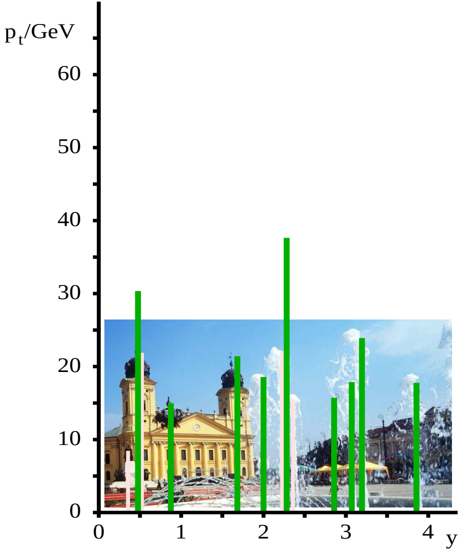
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



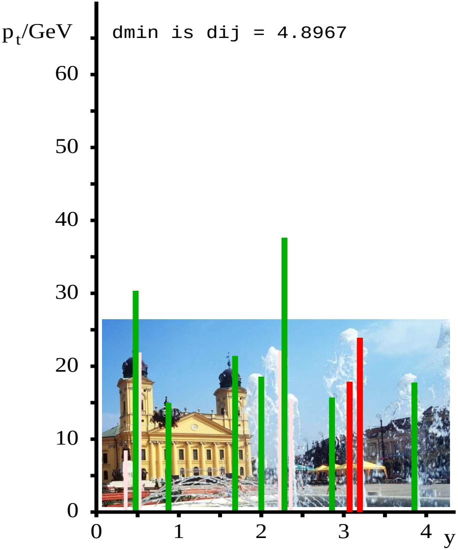
k_t alg.: Find smallest of

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

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



$\mathbf{k_t}$ alg.: Find smallest of

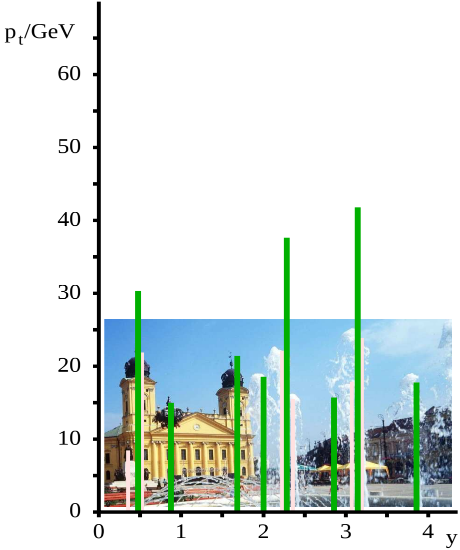
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

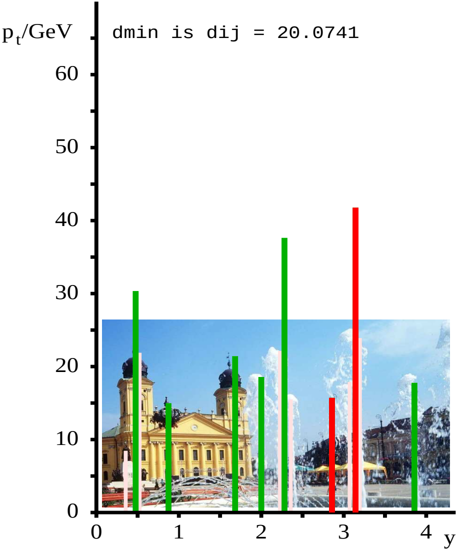
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



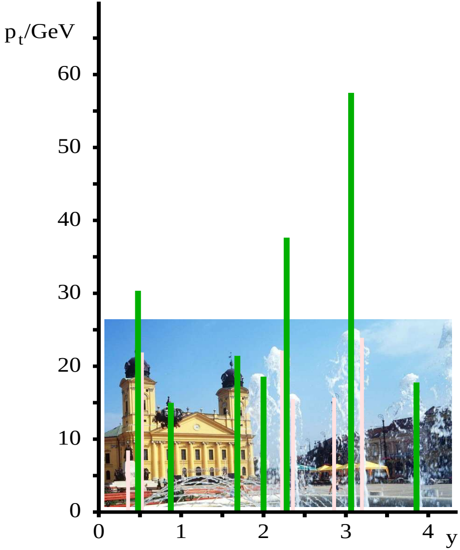
$\mathbf{k_t}$ alg.: Find smallest of

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

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



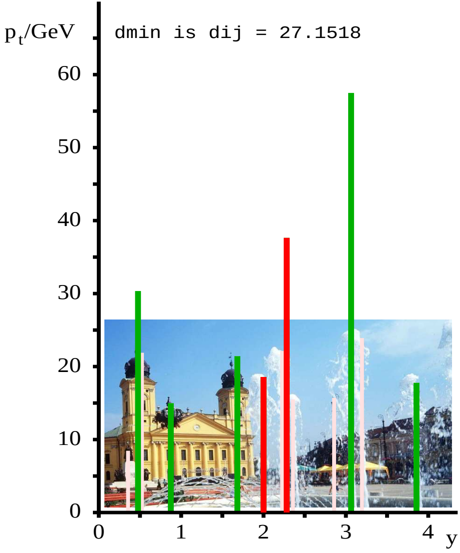
k_t alg.: Find smallest of

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

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

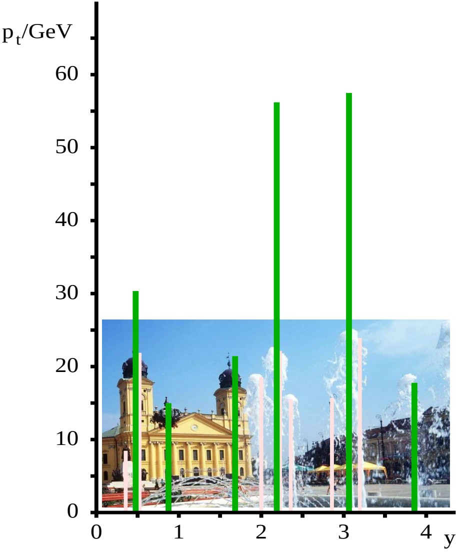
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

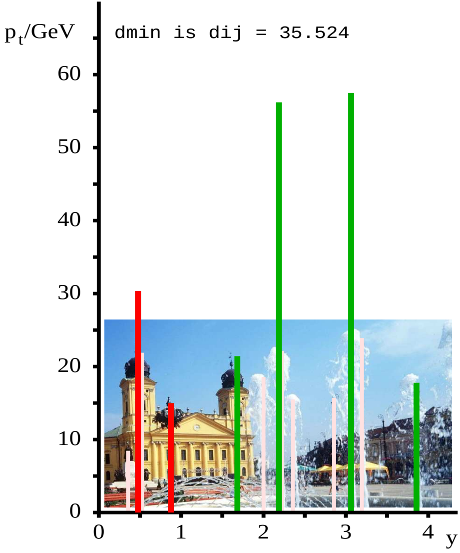
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

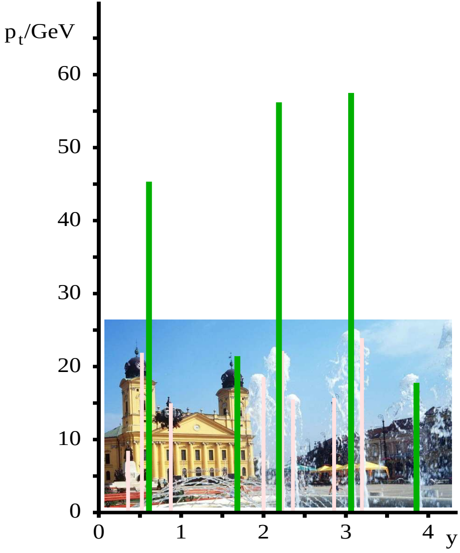
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



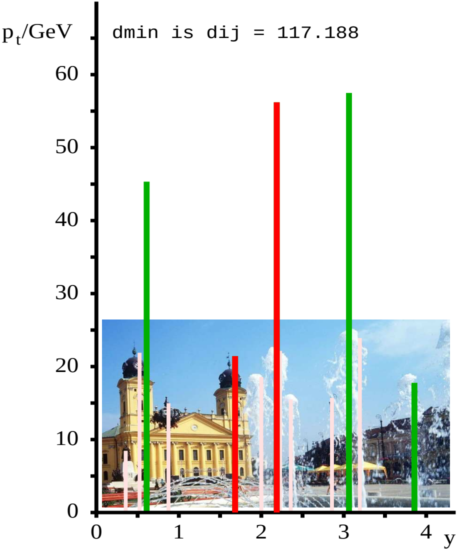
k_t alg.: Find smallest of

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

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



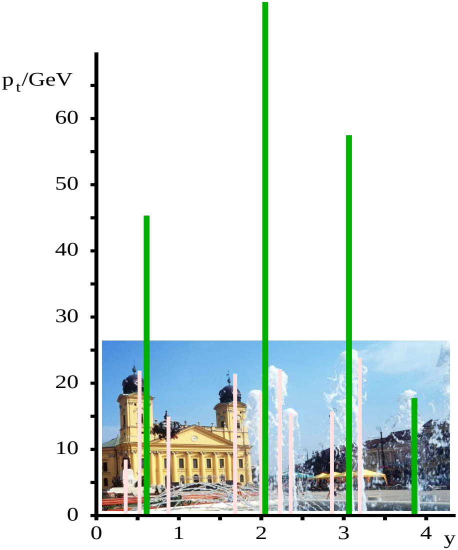
k_t alg.: Find smallest of

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

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



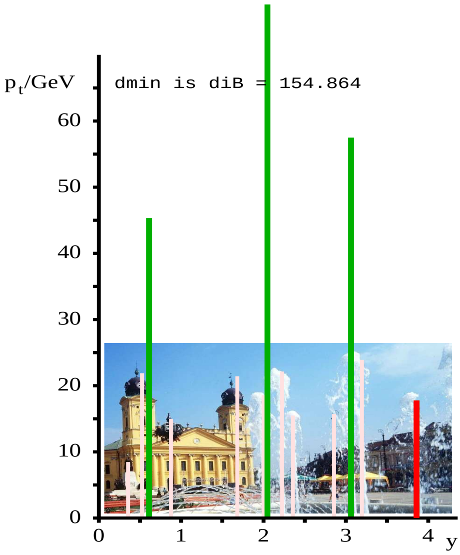
k_t alg.: Find smallest of

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

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

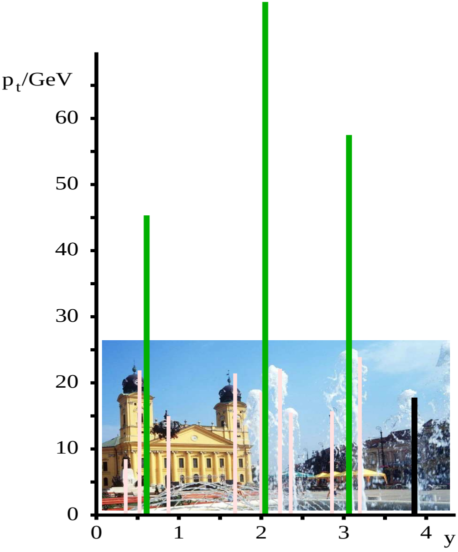
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



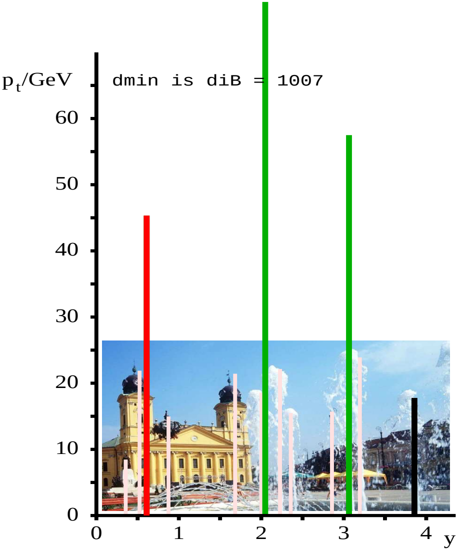
k_t alg.: Find smallest of

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

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



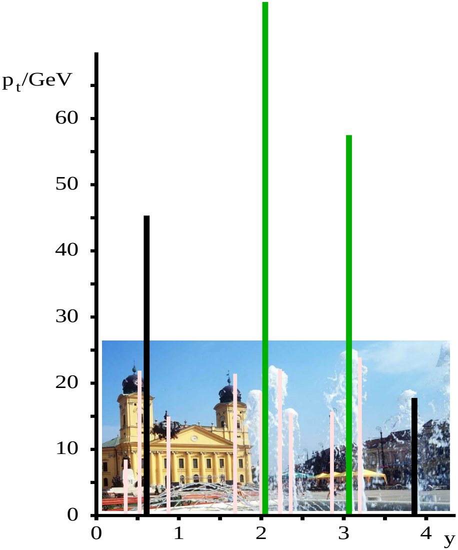
$\mathbf{k_t}$ alg.: Find smallest of

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

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

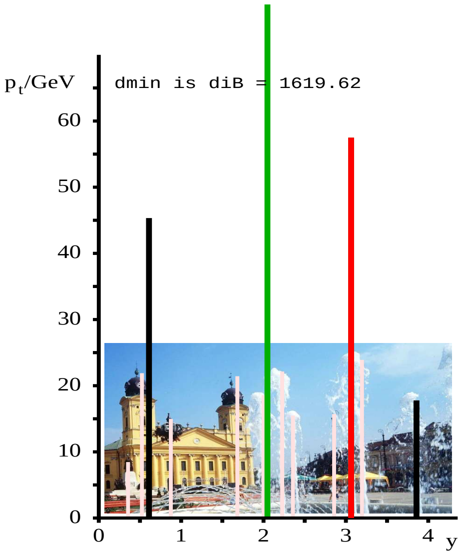
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

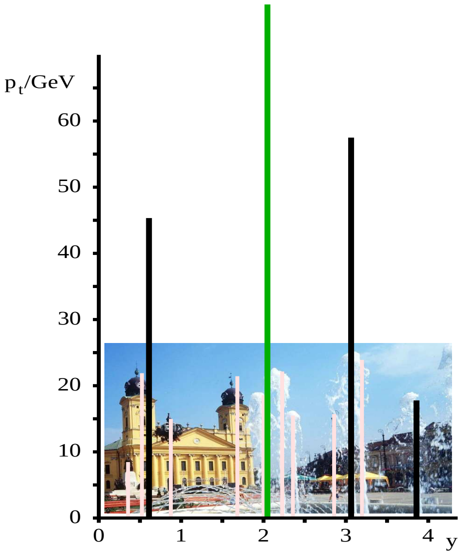
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

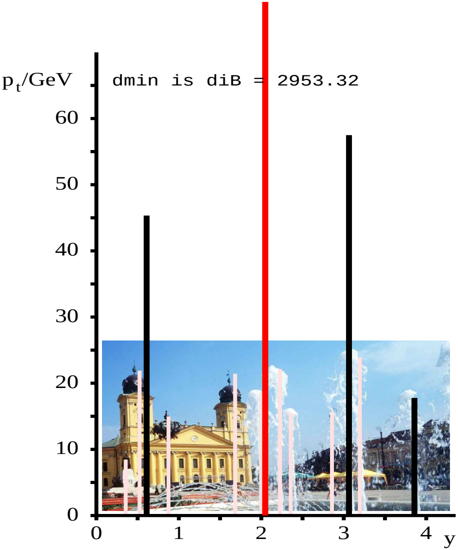
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



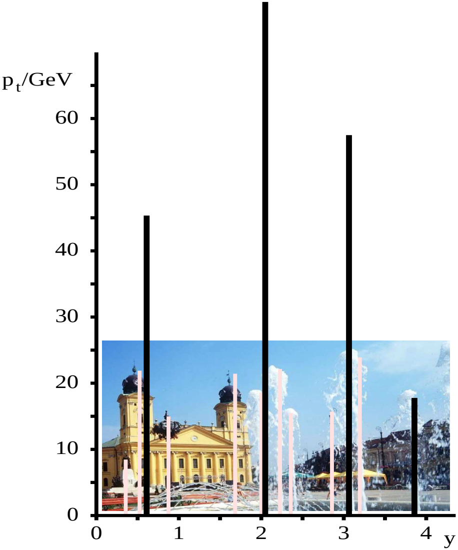
k_t alg.: Find smallest of

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

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



$\mathbf{k_t}$ 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 ⇔ 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 ⇔ 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 ⇔ 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)

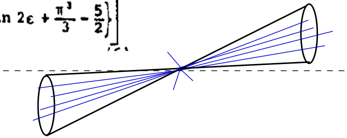
Cone algorithms

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-

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



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

Processing / Finding cones	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

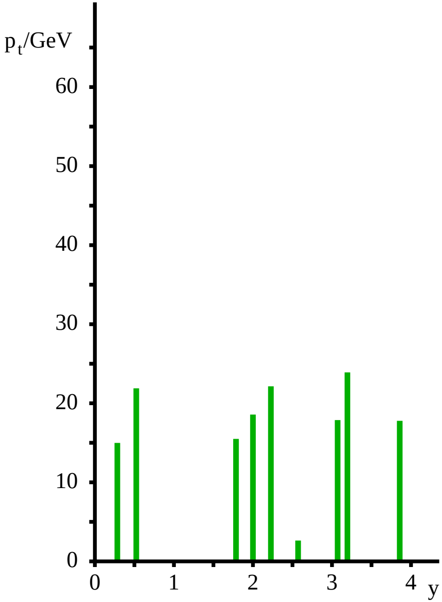
- Cones are always understood as circles in rapidity (y) and azimuth ϕ.
- A particle i is within the cone of radius R around the axis a if

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

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

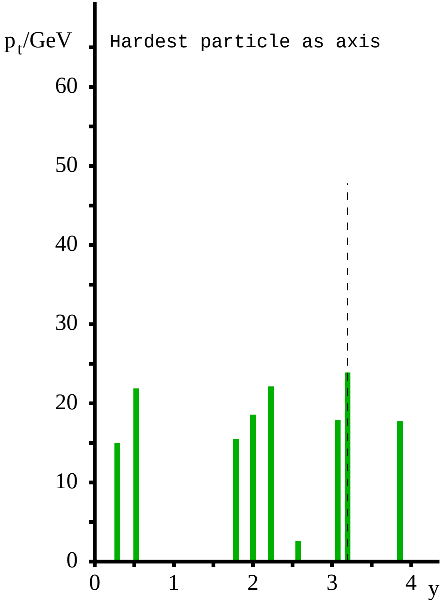
Used e.g. BSM theory; Alpgen MLM

- 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

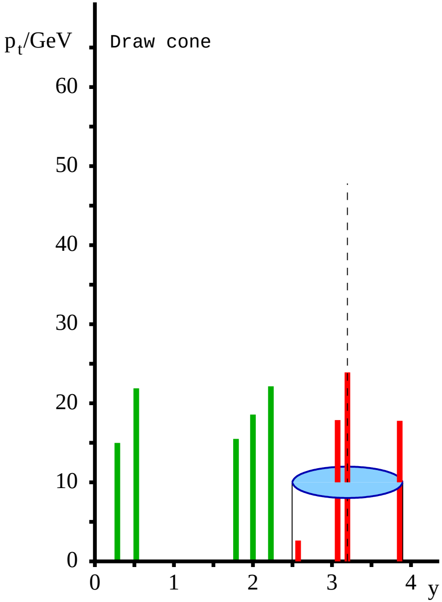
Used e.g. BSM theory; Alpgen MLM

- 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

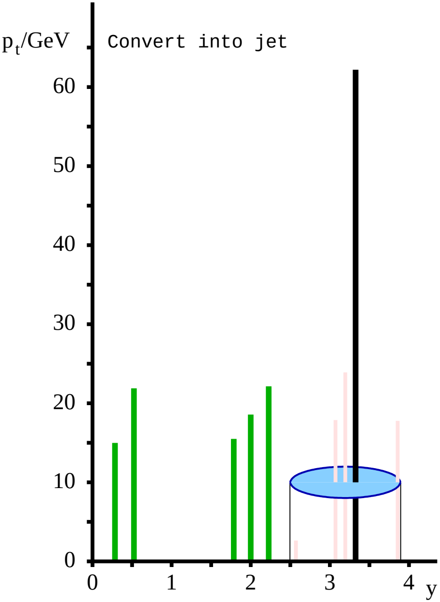
Used e.g. BSM theory; Alpgen MLM

- 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

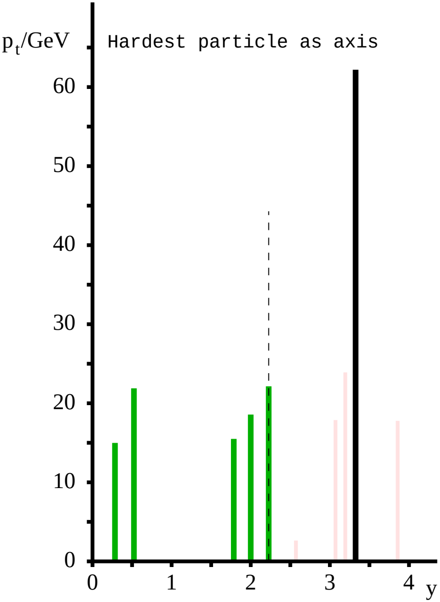
Used e.g. BSM theory; Alpgen MLM

- 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

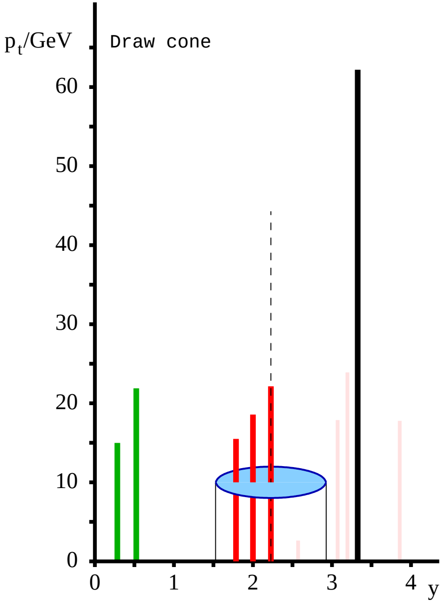
Used e.g. BSM theory; Alpgen MLM

- 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)



<u>The simplest of the cones</u>

PyCell, CellJet, GetJet

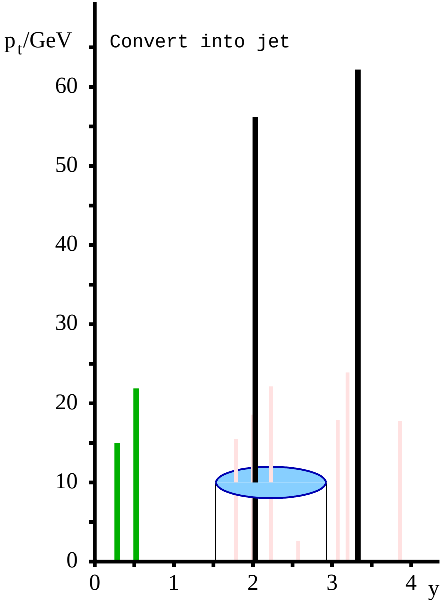
Used e.g. BSM theory; Alpgen MLM

- 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

<u>Notes</u>

- “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

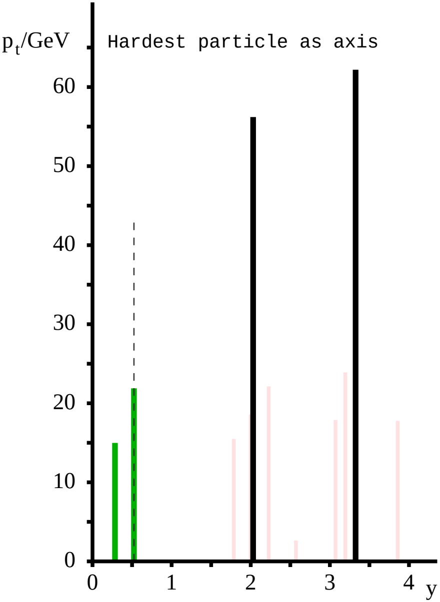
Used e.g. BSM theory; Alpgen MLM

- 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

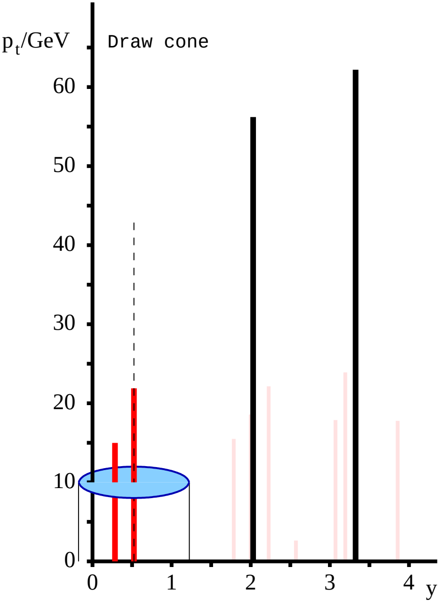
Used e.g. BSM theory; Alpgen MLM

- 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

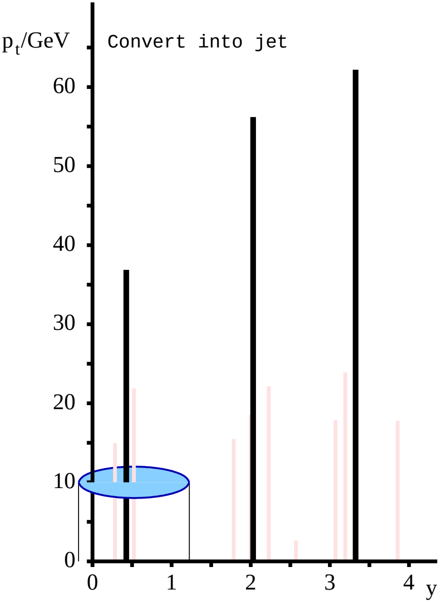
Used e.g. BSM theory; Alpgen MLM

- 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

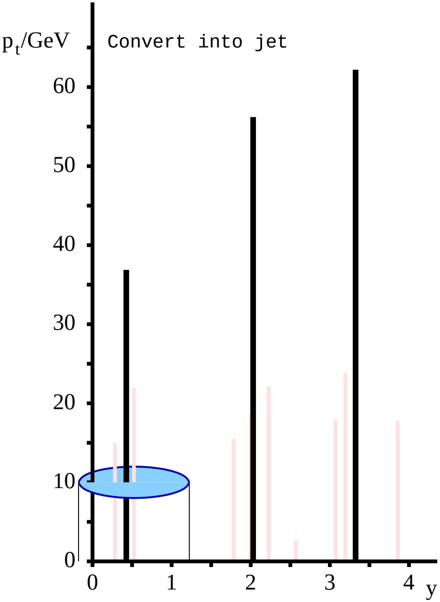
Used e.g. BSM theory; Alpgen MLM

- 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

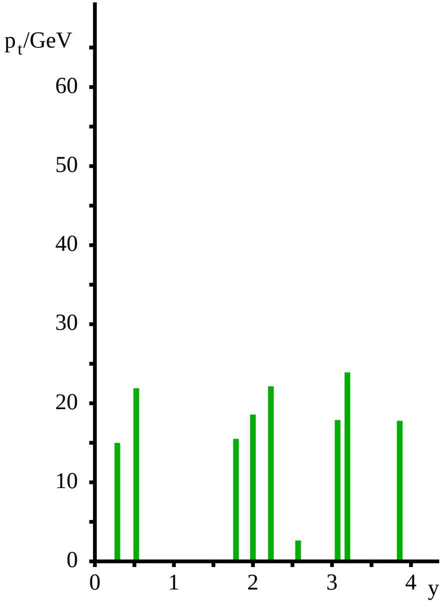
Used e.g. BSM theory; Alpgen MLM

- 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

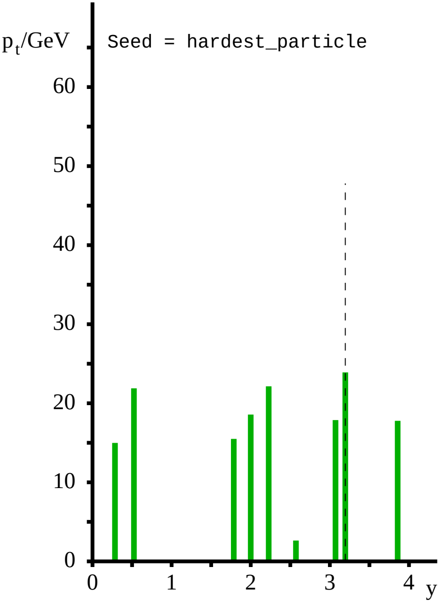
e.g. CMS iterative cone

- 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

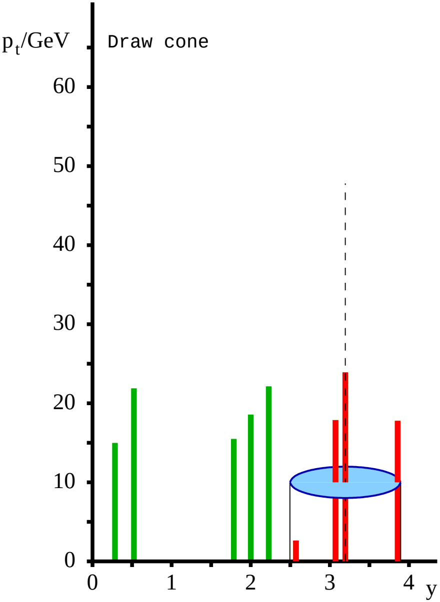
e.g. CMS iterative cone

- 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

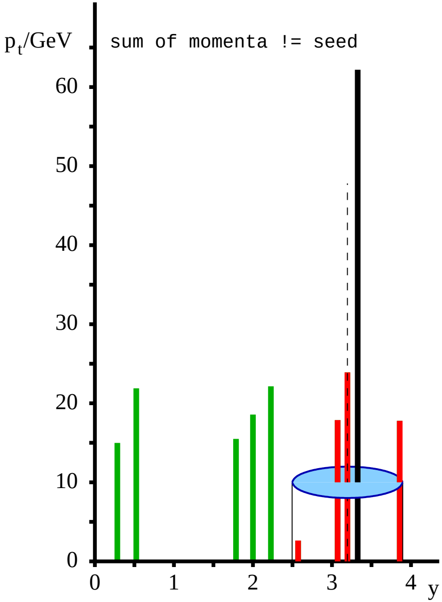
e.g. CMS iterative cone

- 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

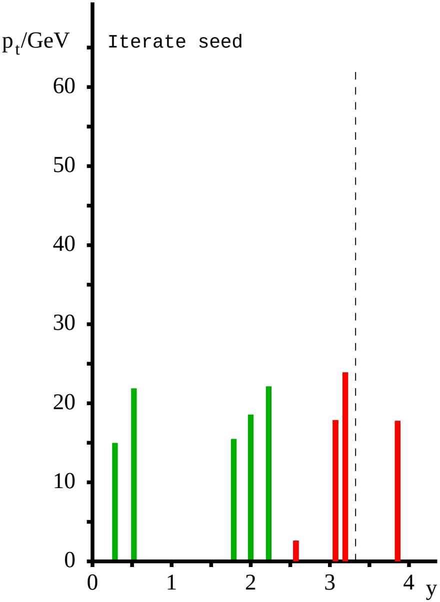
e.g. CMS iterative cone

- 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

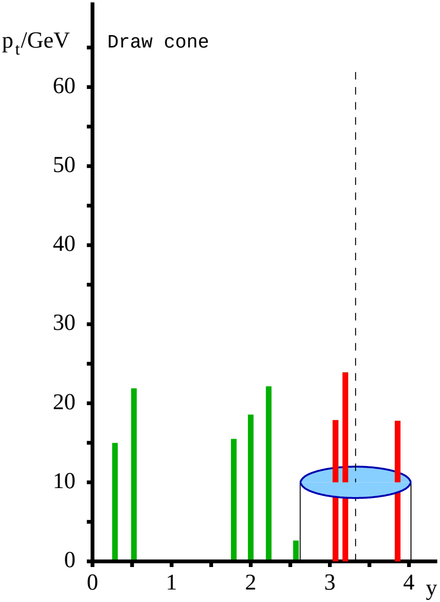
e.g. CMS iterative cone

- 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

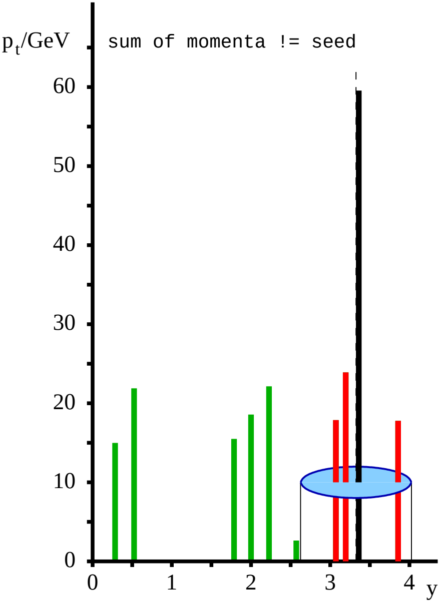
e.g. CMS iterative cone

- 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

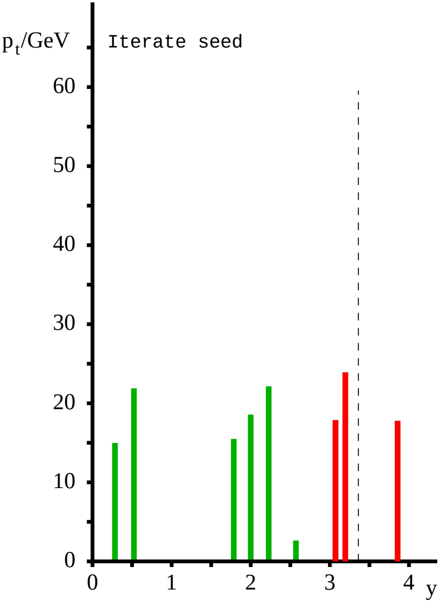
e.g. CMS iterative cone

- 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

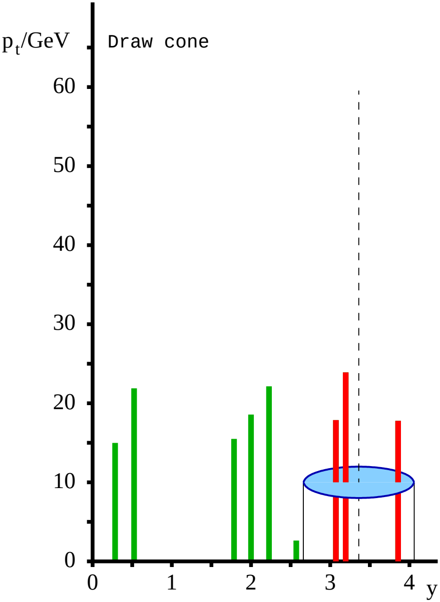
e.g. CMS iterative cone

- 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

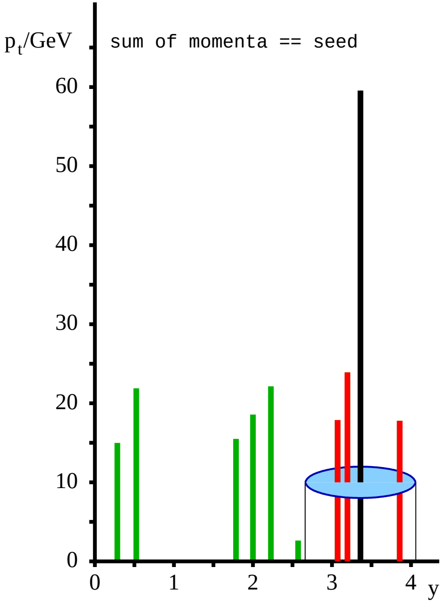
e.g. CMS iterative cone

- 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)



<u>Next-simplest of the cones</u>

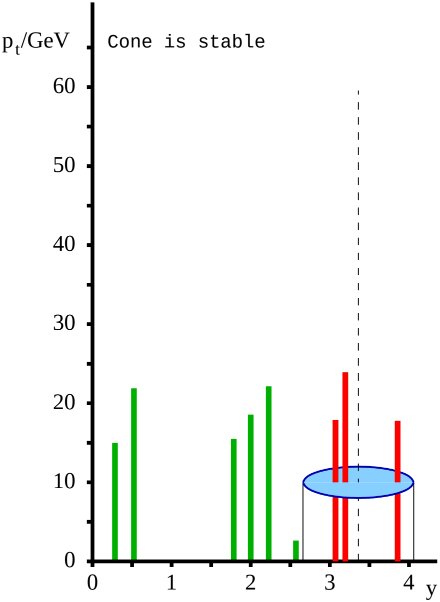
e.g. CMS iterative cone

- 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

<u>Notes</u>

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

Iterative Cone, Prog Removal (IC-PR)

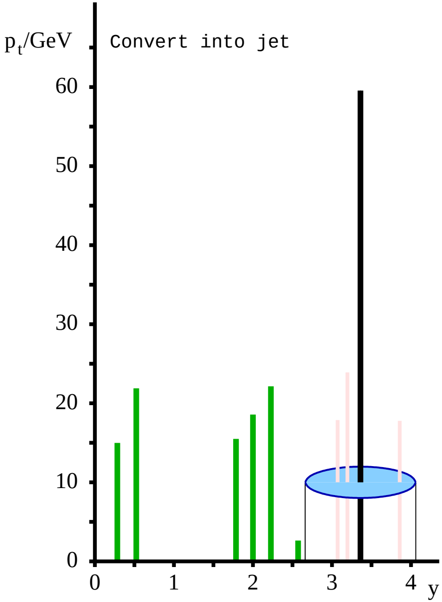


Next-simplest of the cones

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

Iterative Cone, Prog Removal (IC-PR)

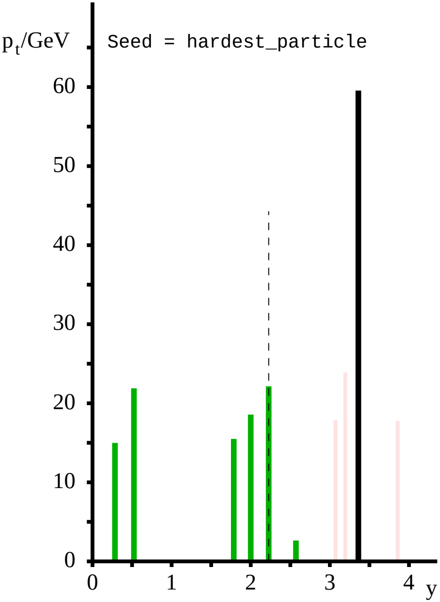


Next-simplest of the cones

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

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

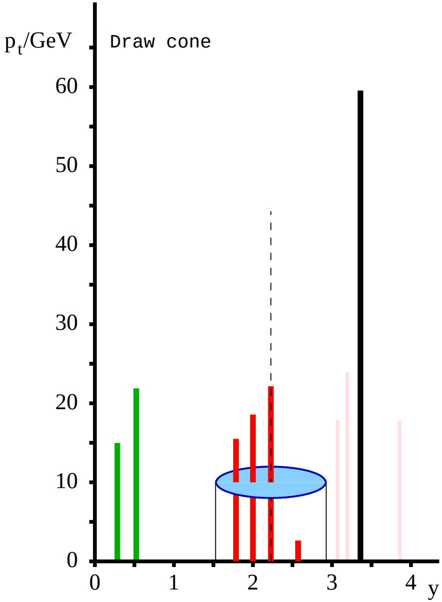
e.g. CMS iterative cone

- 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

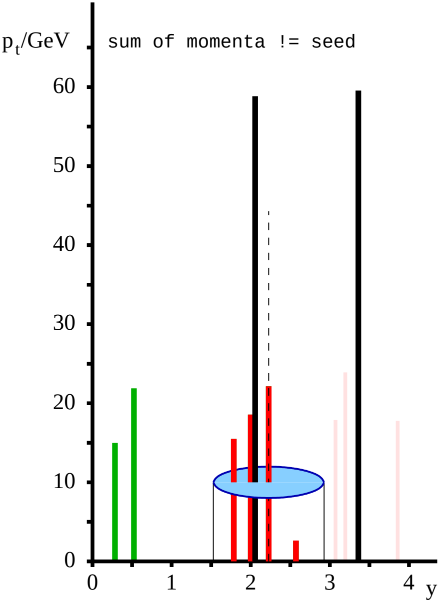
e.g. CMS iterative cone

- 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)



<u>Next-simplest of the cones</u>

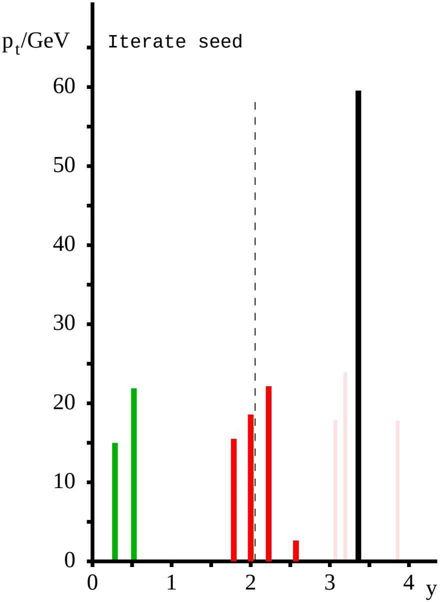
e.g. CMS iterative cone

- 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

<u>Notes</u>

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

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

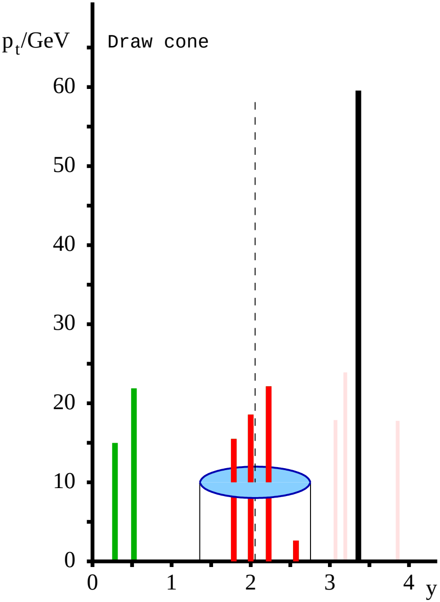
e.g. CMS iterative cone

- 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)

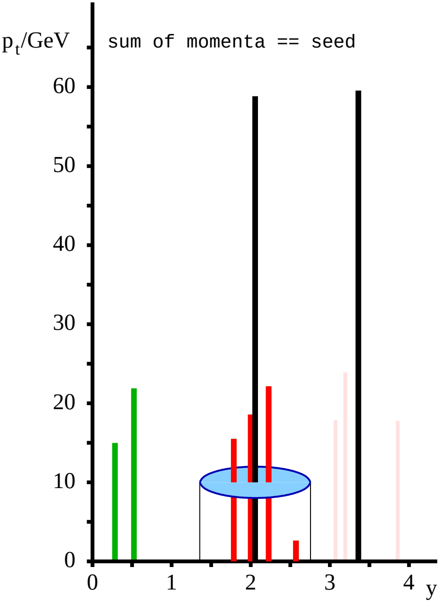


Next-simplest of the cones

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

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

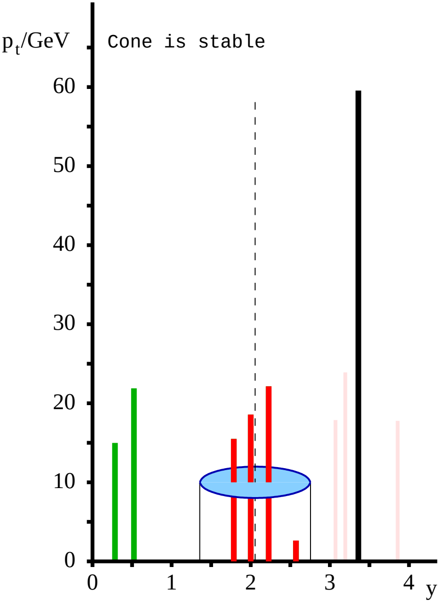
e.g. CMS iterative cone

- 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

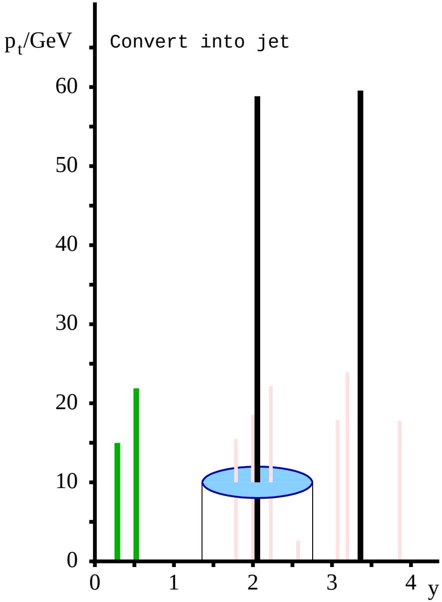
e.g. CMS iterative cone

- 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

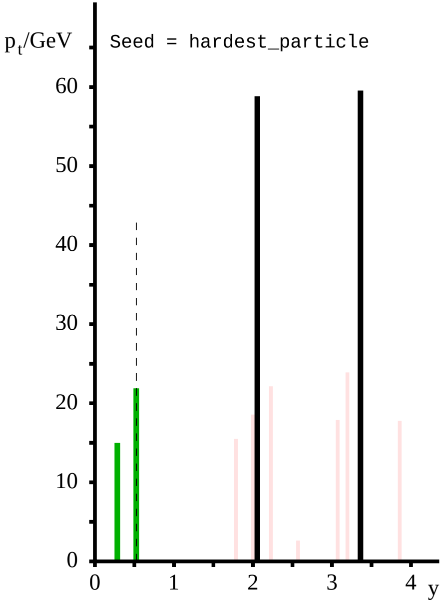
e.g. CMS iterative cone

- 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)



<u>Next-simplest of the cones</u>

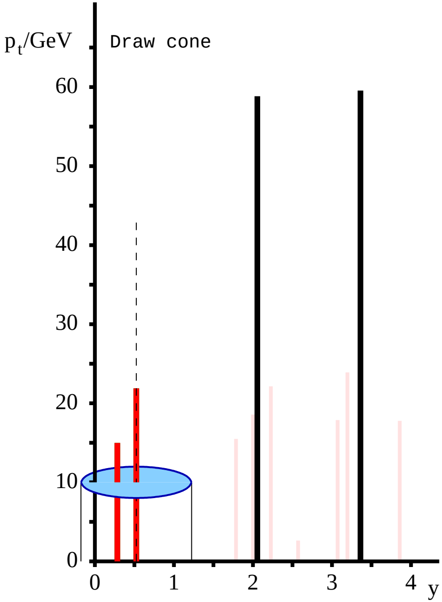
e.g. CMS iterative cone

- 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

<u>Notes</u>

- “Hardest particle” is collinear unsafe more right away...

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

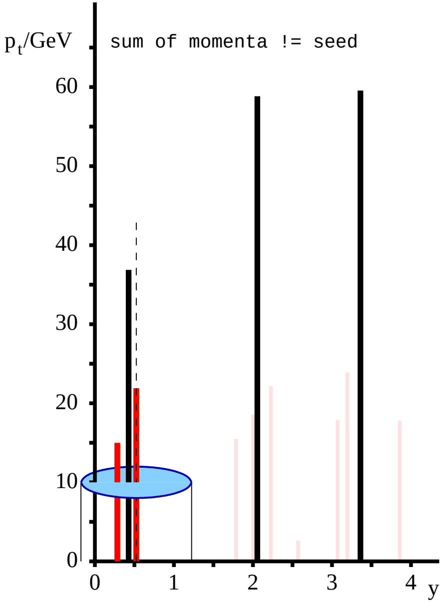
e.g. CMS iterative cone

- 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

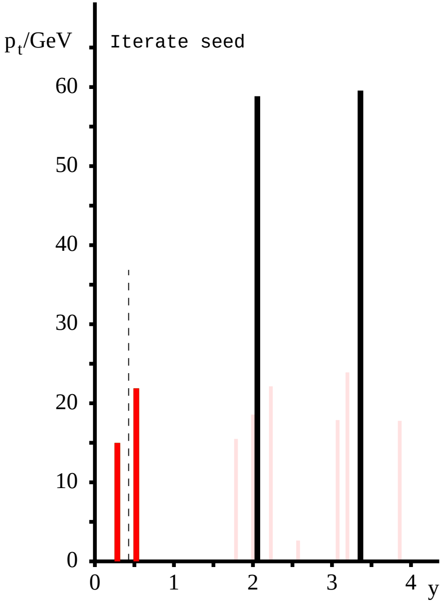
e.g. CMS iterative cone

- 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

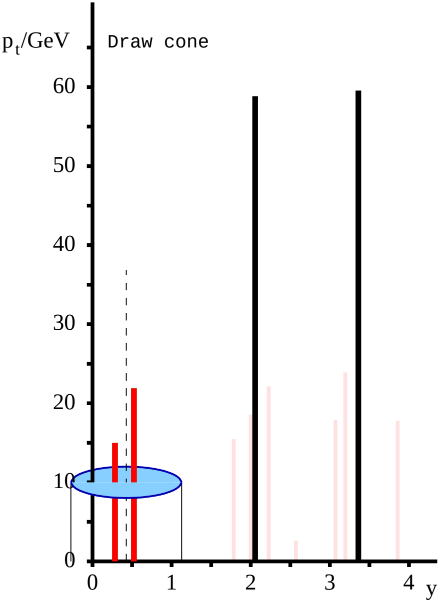
e.g. CMS iterative cone

- 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)

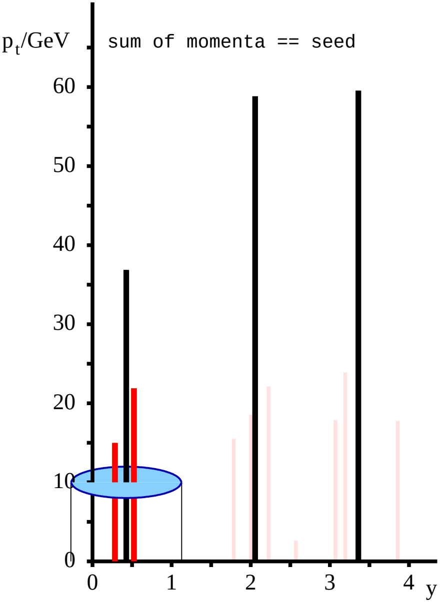


Next-simplest of the cones

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

Iterative Cone, Prog Removal (IC-PR)



Next-simplest of the cones

e.g. CMS iterative cone

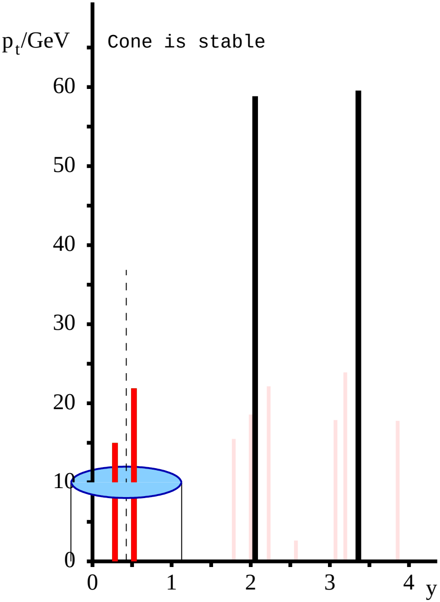
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

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

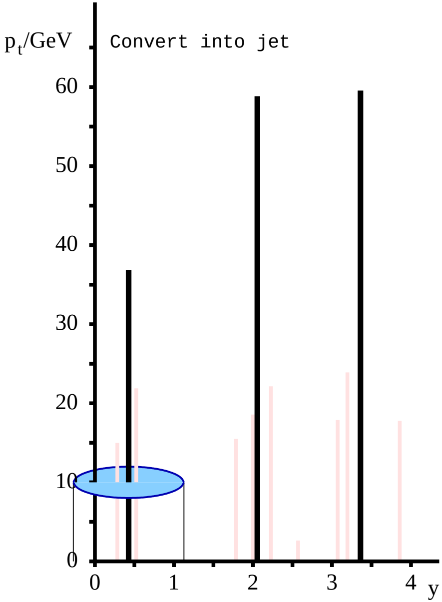
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

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

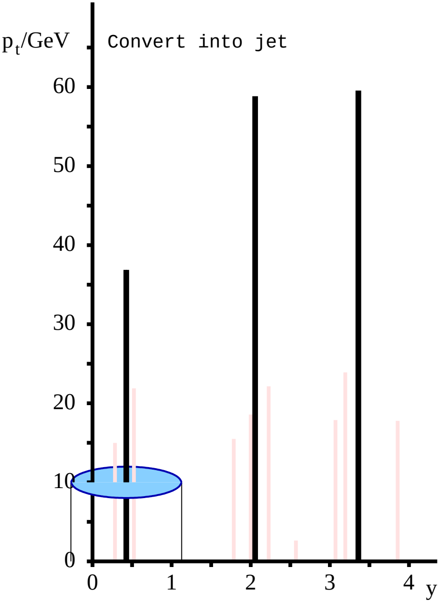
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

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

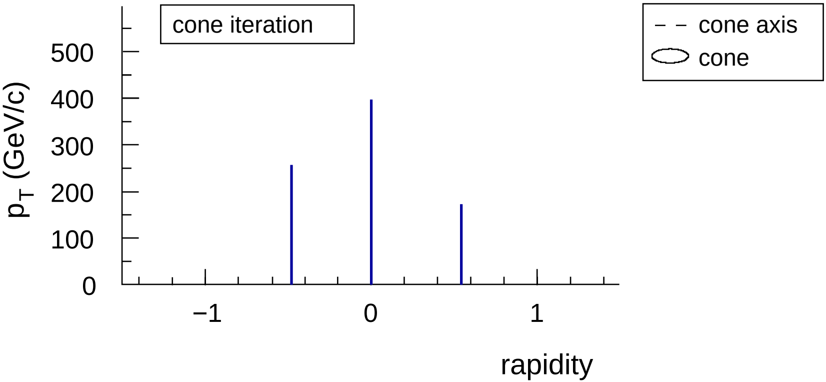
- Take hardest particle as seed for cone axis
- Draw cone around seed
- Sum the momenta use as new seed direction, iterate until stable
- Convert contents into a "jet" and remove from event

Notes

- "Hardest particle" is collinear unsafe

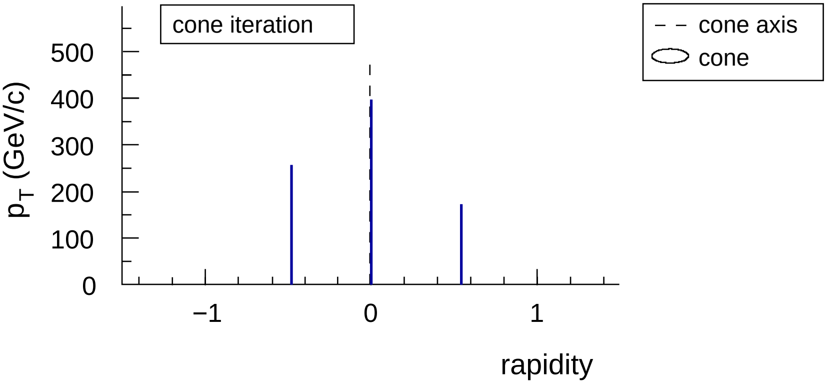
more right away...

ICPR iteration issue



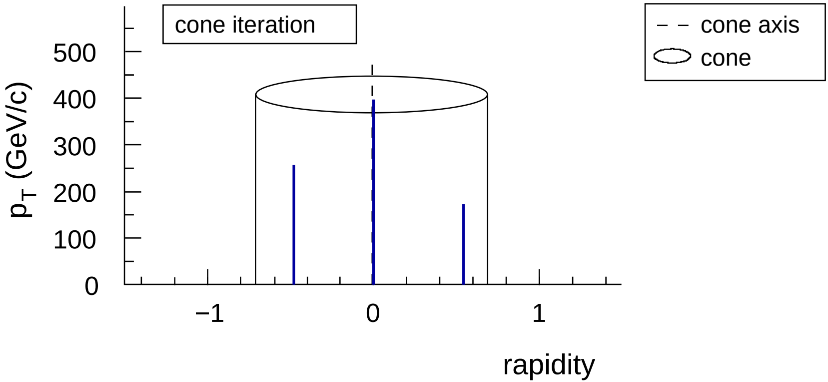
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



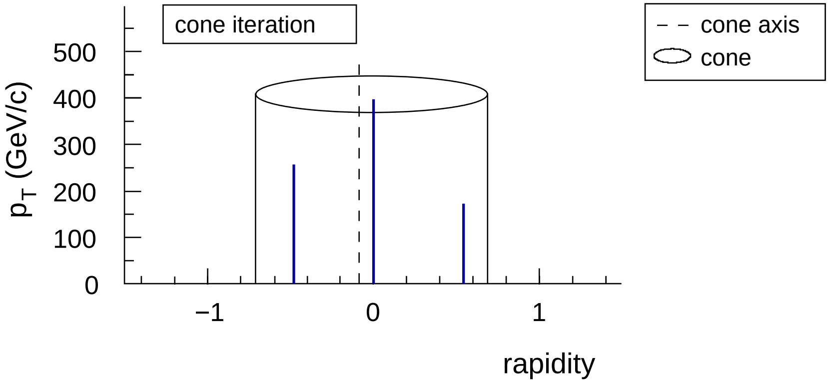
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



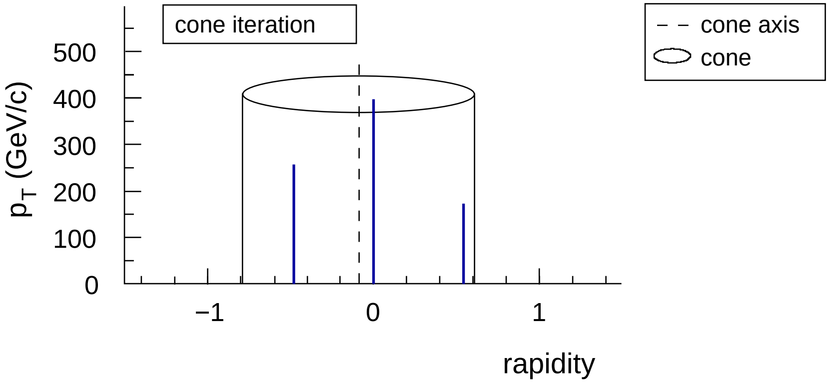
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



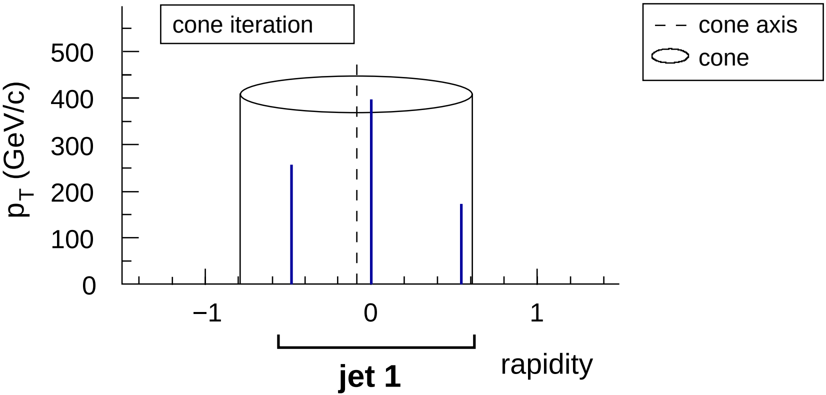
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



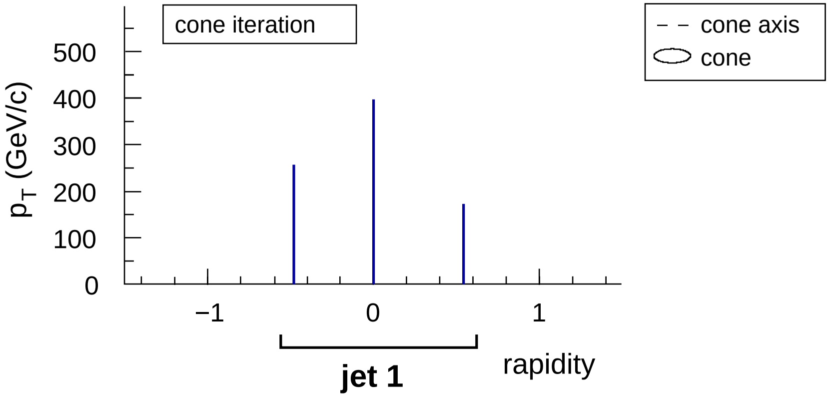
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



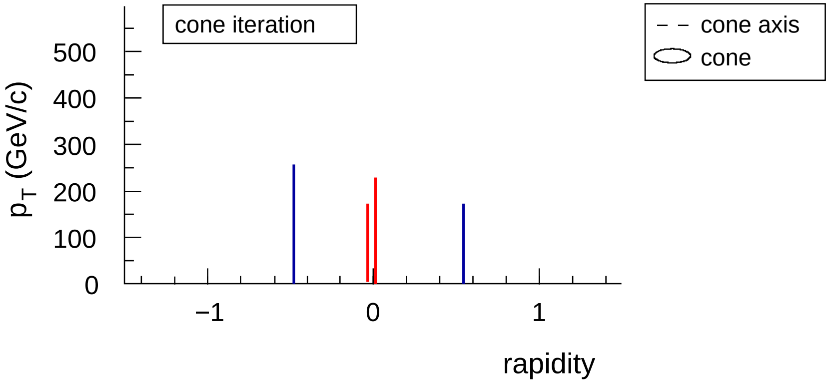
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



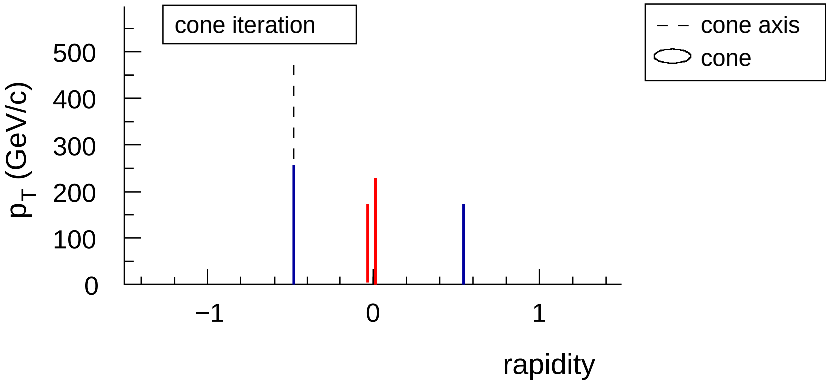
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



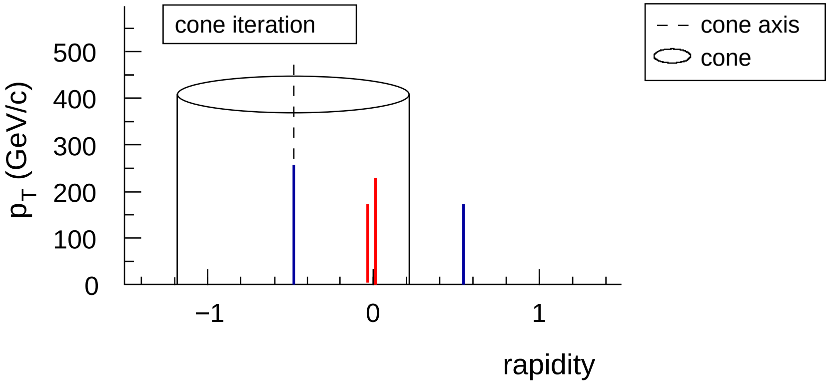
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



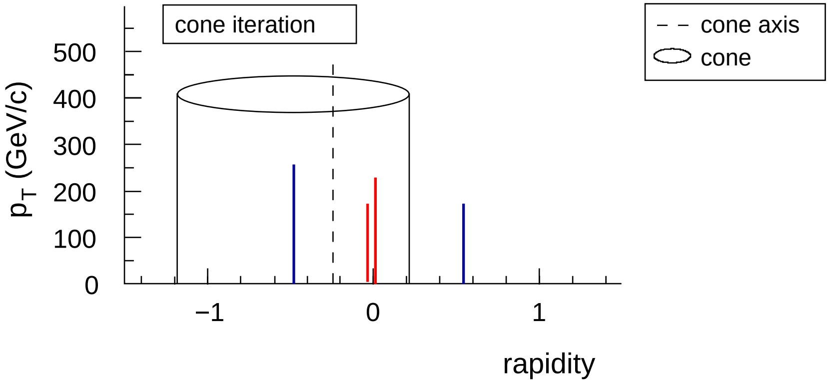
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



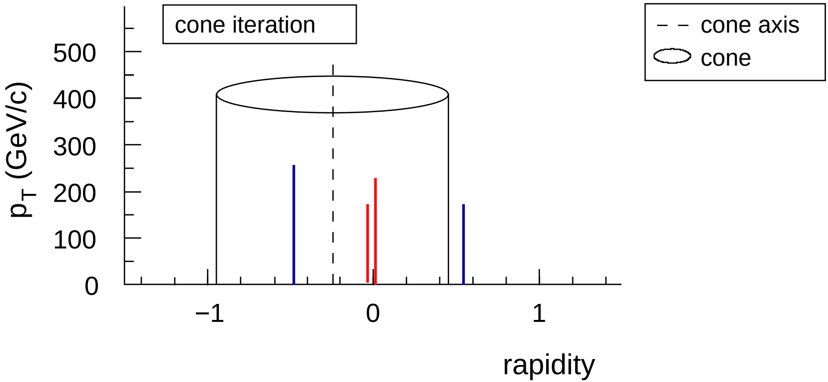
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



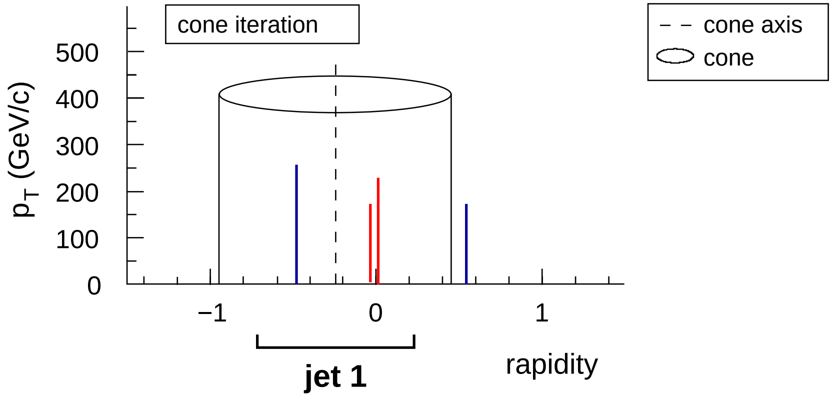
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



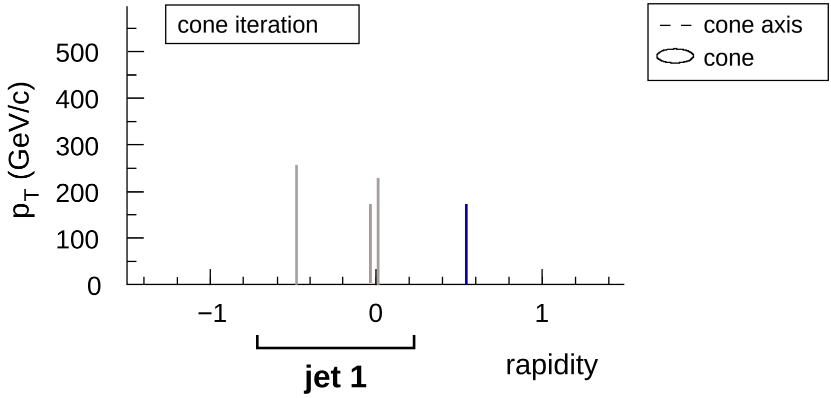
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



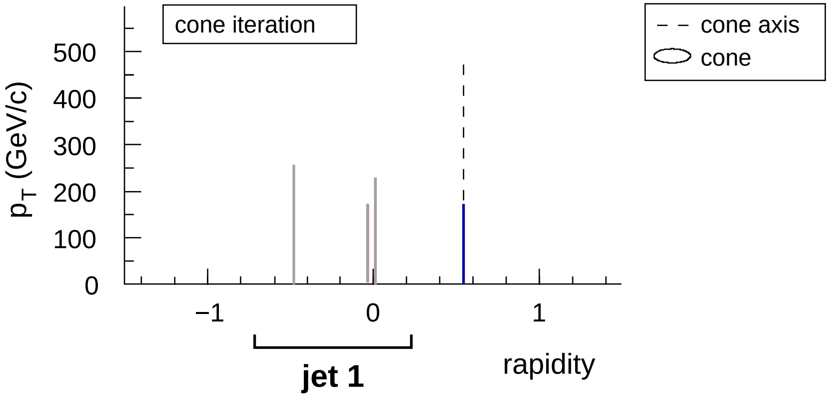
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



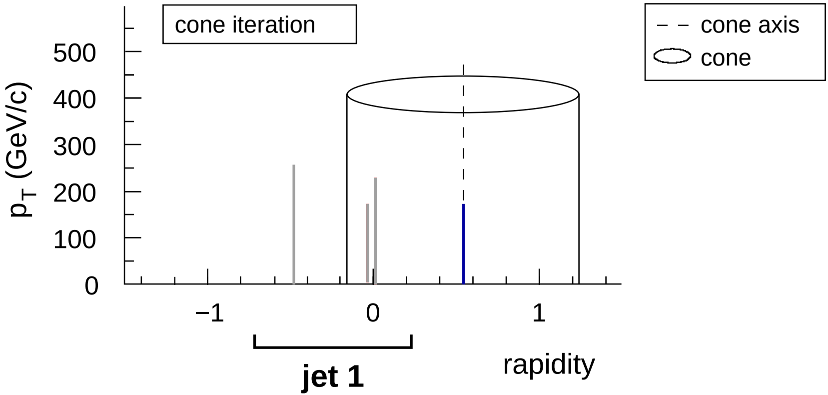
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



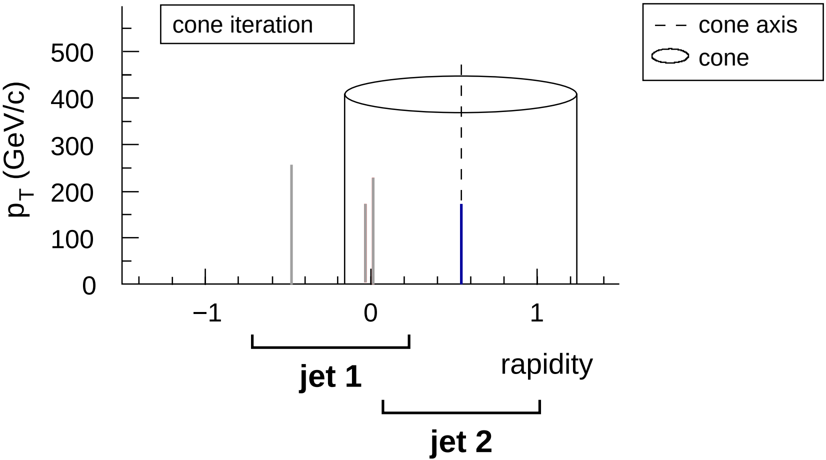
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



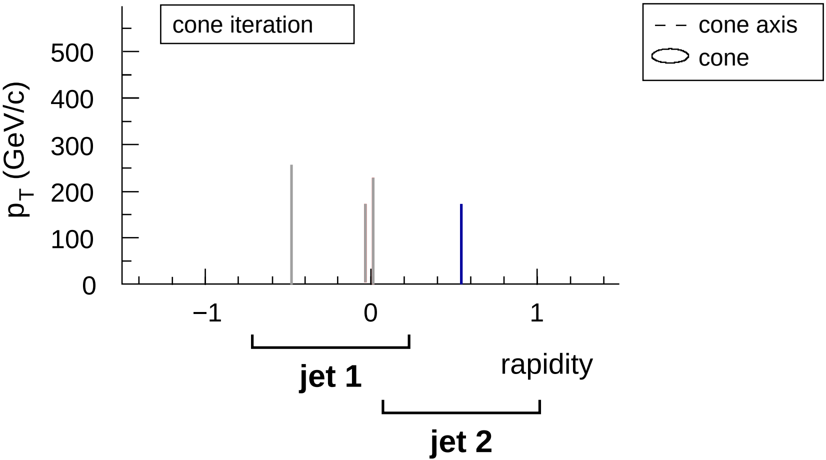
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe $\Longrightarrow$ perturbative calculations give ∞

ICPR iteration issue



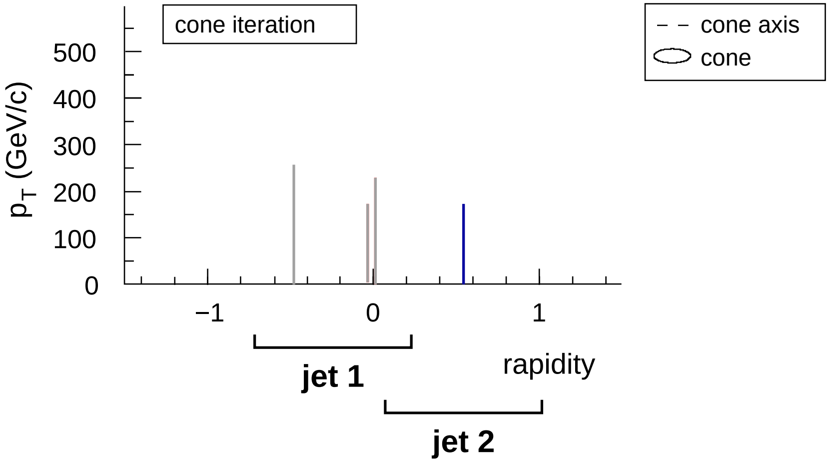
Collinear splitting can modify the hard jets: ICPR algorithms are collinear unsafe ⟹ perturbative calculations give ∞

ICPR iteration issue



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

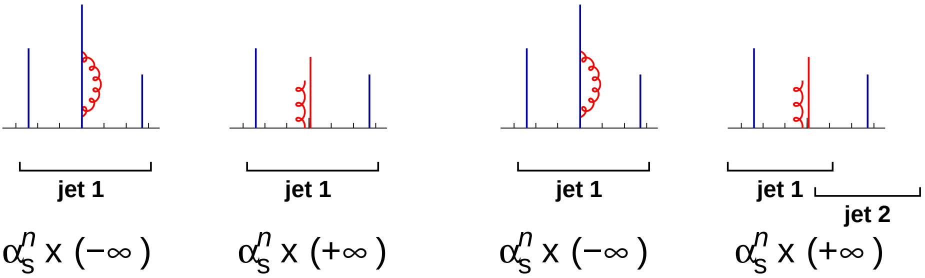
ICPR iteration issue



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

Consequences of collinear unsafety

Collinear Safe | **Collinear Unsafe**

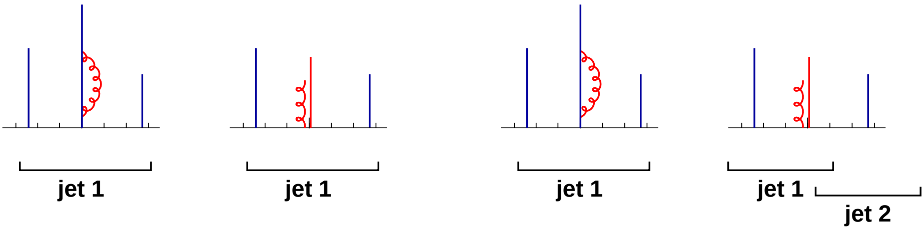


Infinities cancel | **Infinities do not cancel**

Invalidates perturbation theory

Consequences of collinear unsafety

Collinear Safe | **Collinear Unsafe**



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

Infinities cancel | **Infinities do not cancel**

Invalidates perturbation theory

[End of lecture 1]

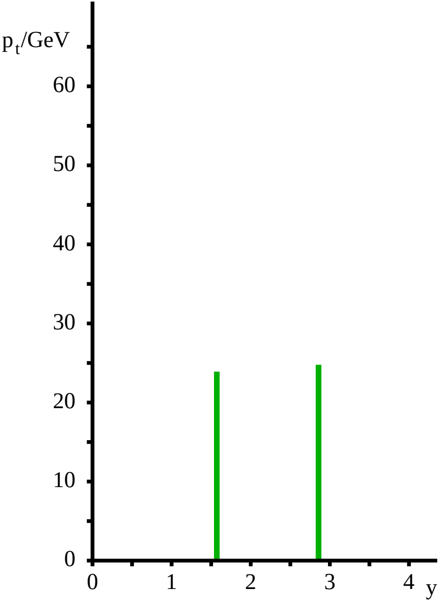
<u>So far</u>

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

<u>Tomorrow</u>

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

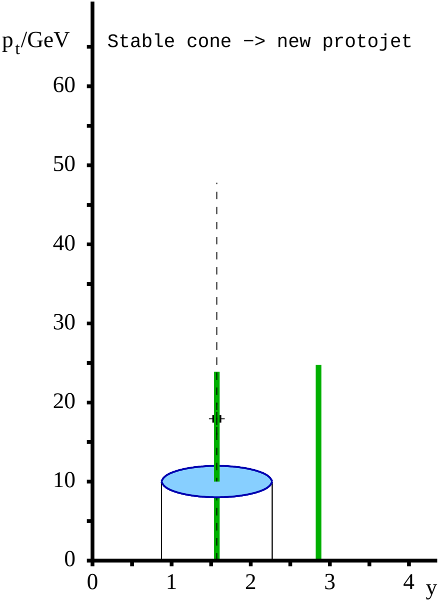
ICSM: infrared safety



(-

Event with extra

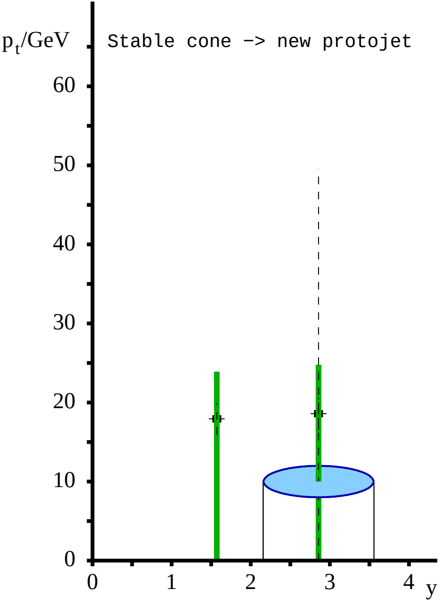
ICSM: infrared safety



(-

Event with extra

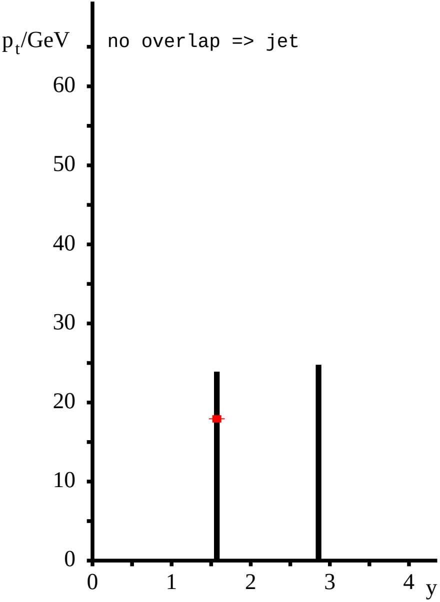
ICSM: infrared safety



(-

Event with extra

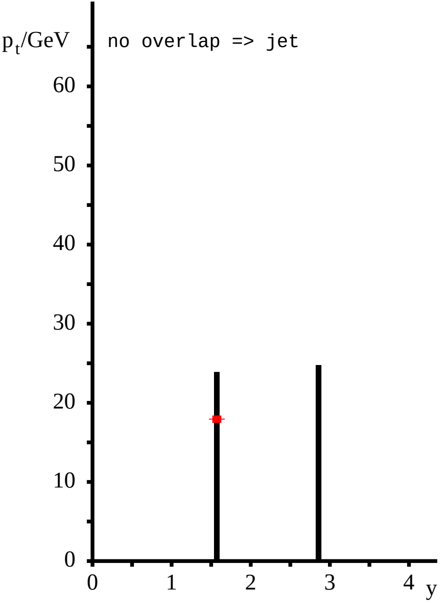
ICSM: infrared safety

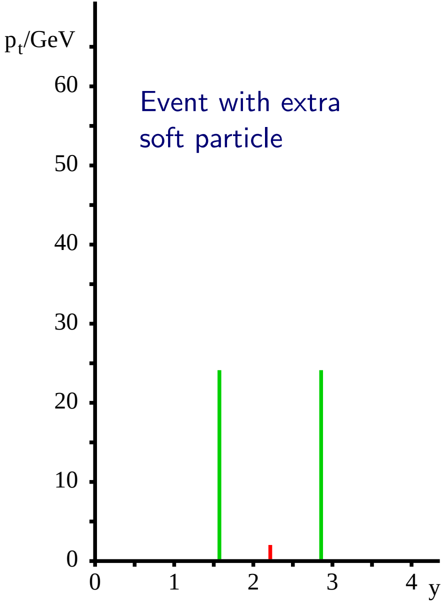


(-

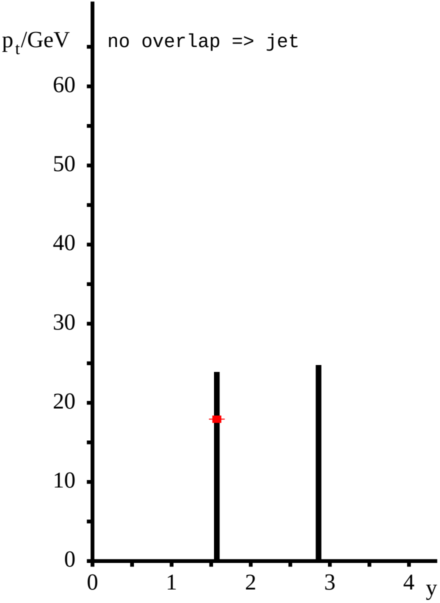
Event with extra

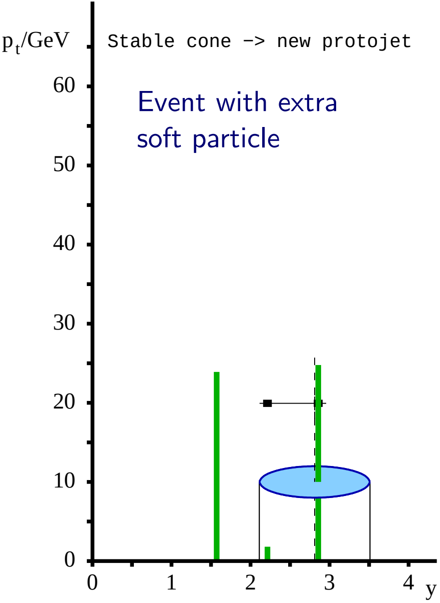
ICSM: infrared safety



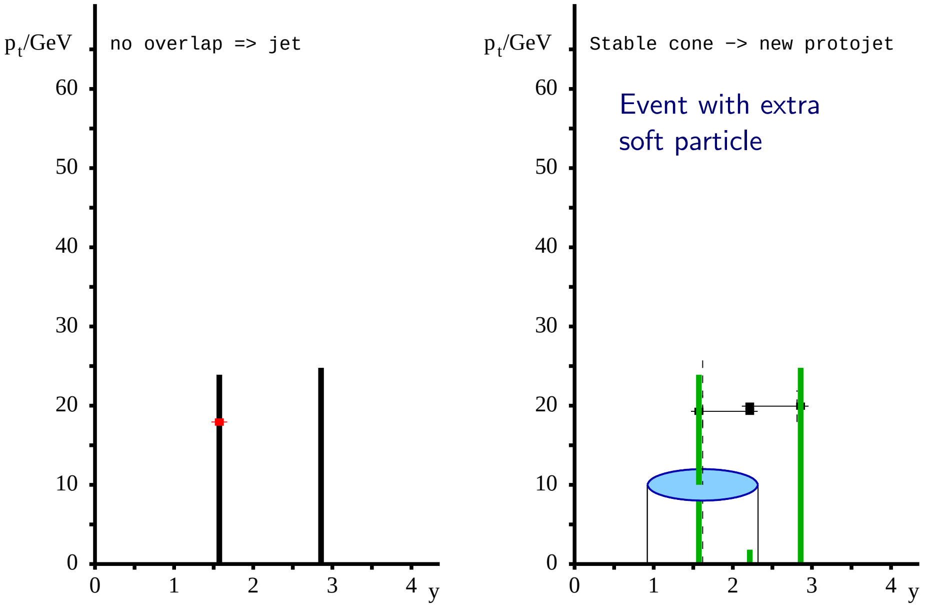


ICSM: infrared safety

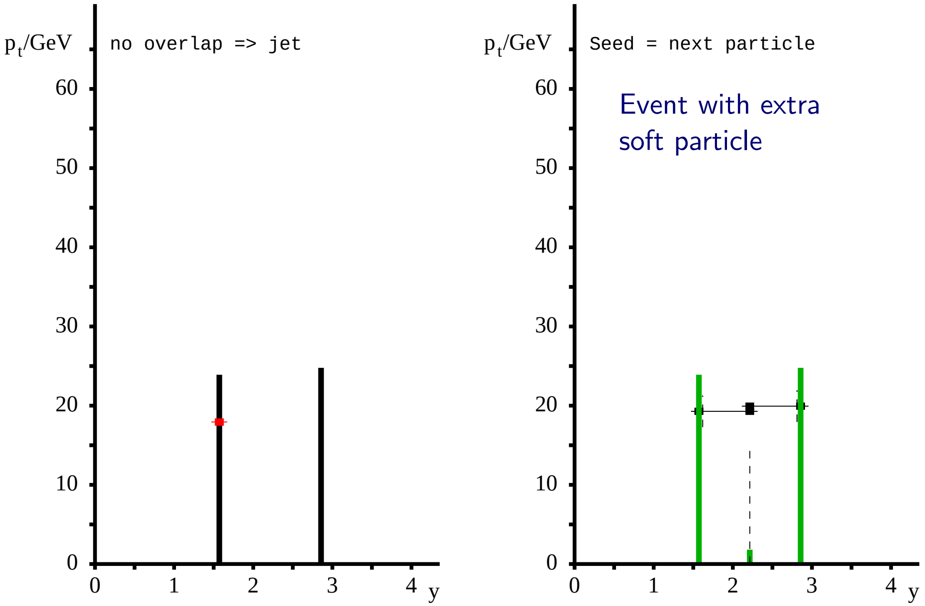




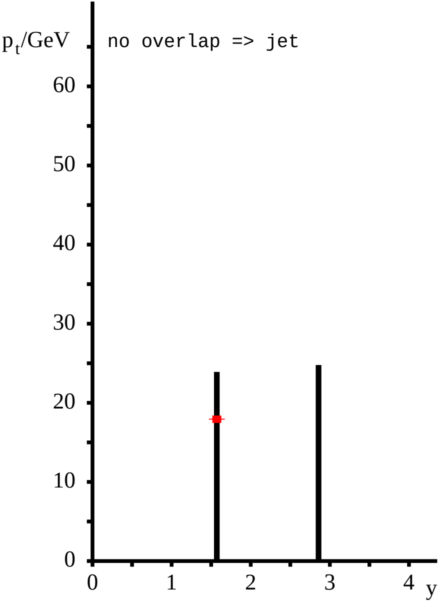
ICSM: infrared safety

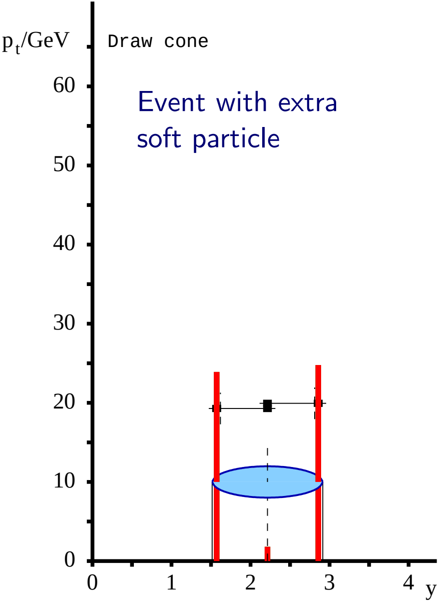


ICSM: infrared safety



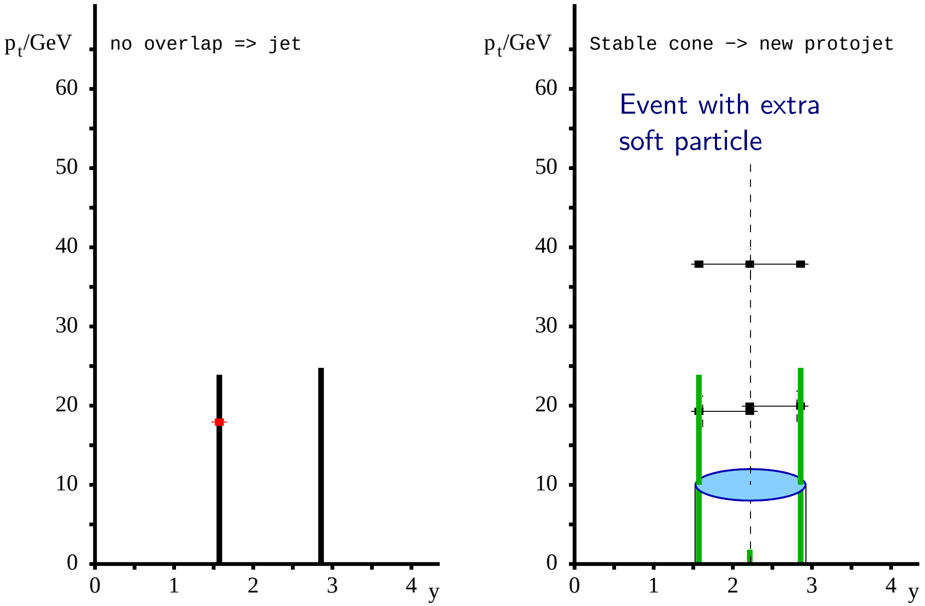
ICSM: infrared safety



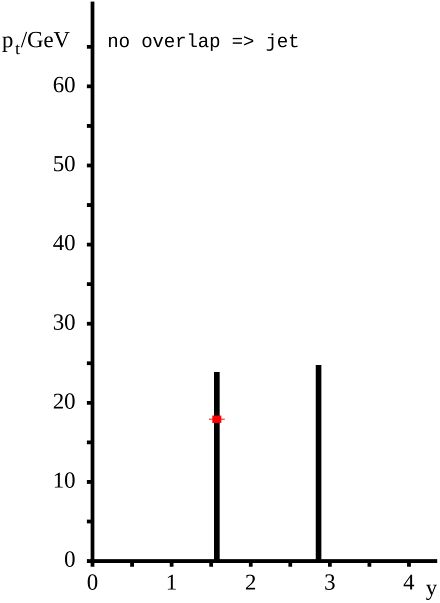


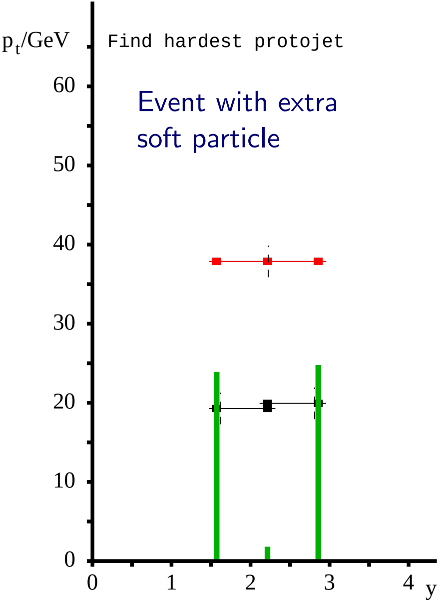
Event with extra soft particle

ICSM: infrared safety



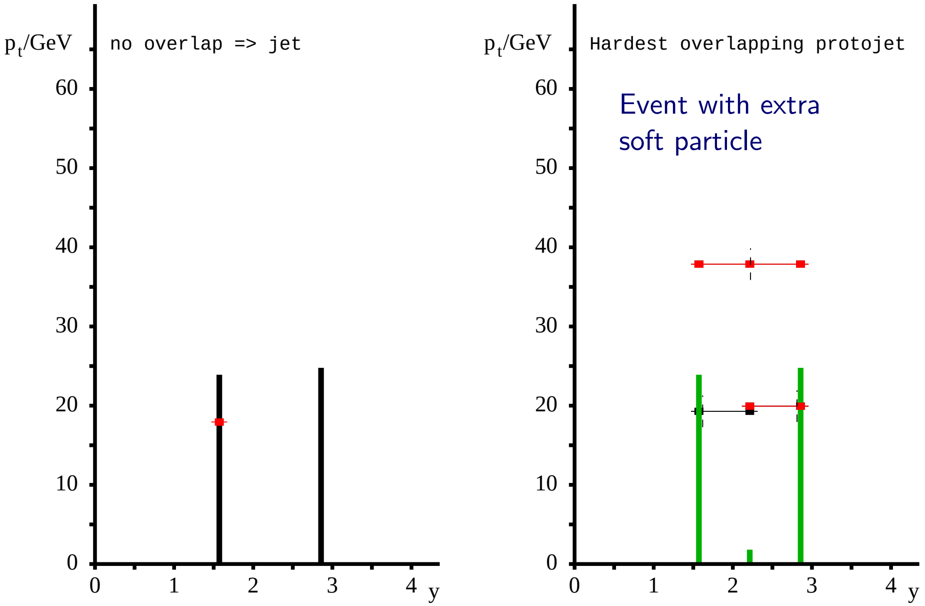
ICSM: infrared safety



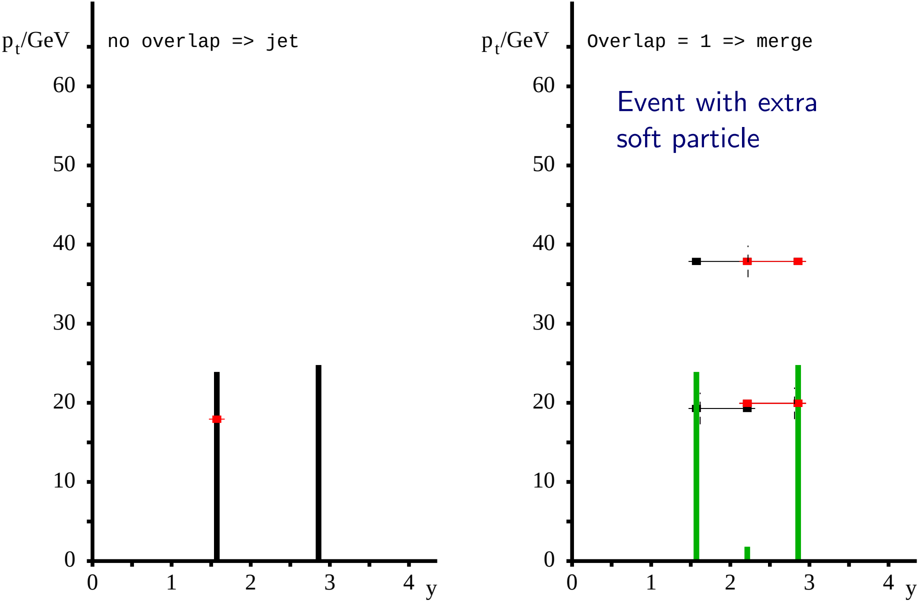


Event with extra soft particle

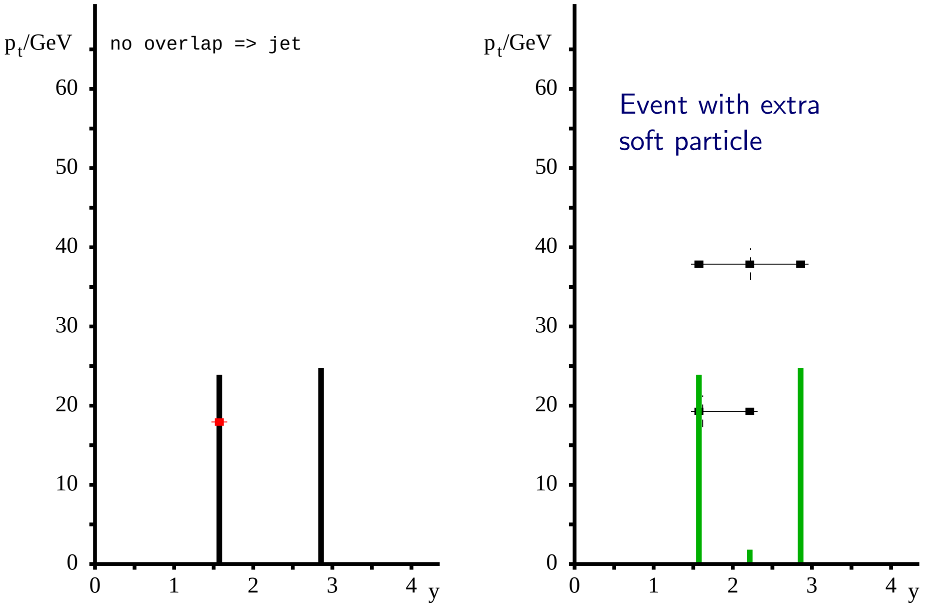
ICSM: infrared safety



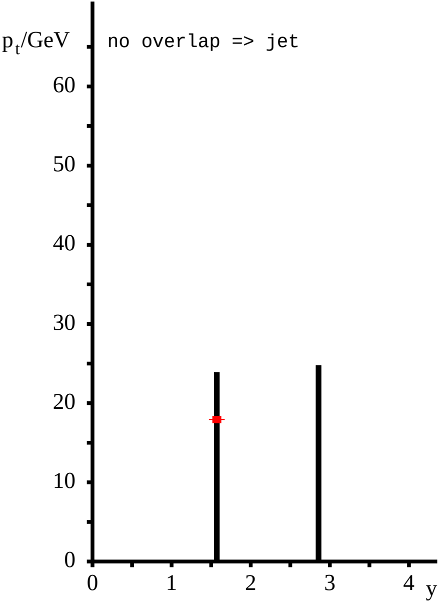
ICSM: infrared safety

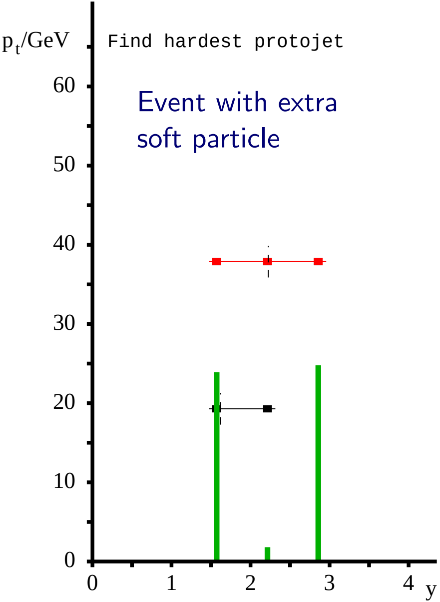


ICSM: infrared safety

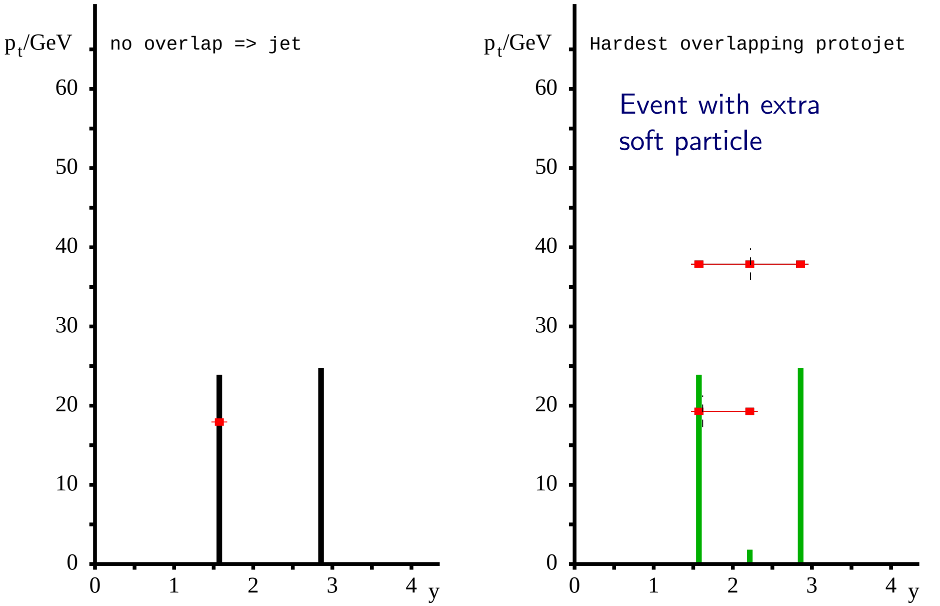


ICSM: infrared safety

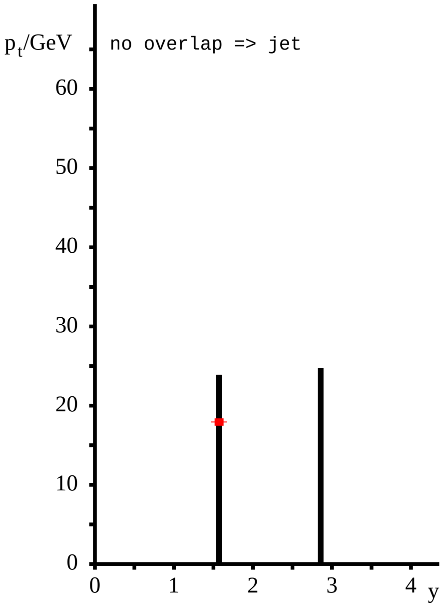


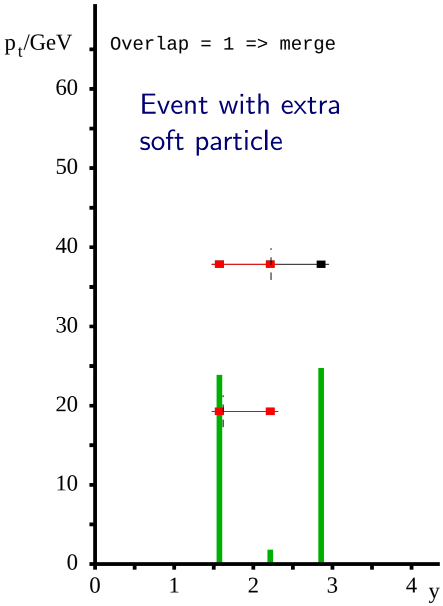


ICSM: infrared safety

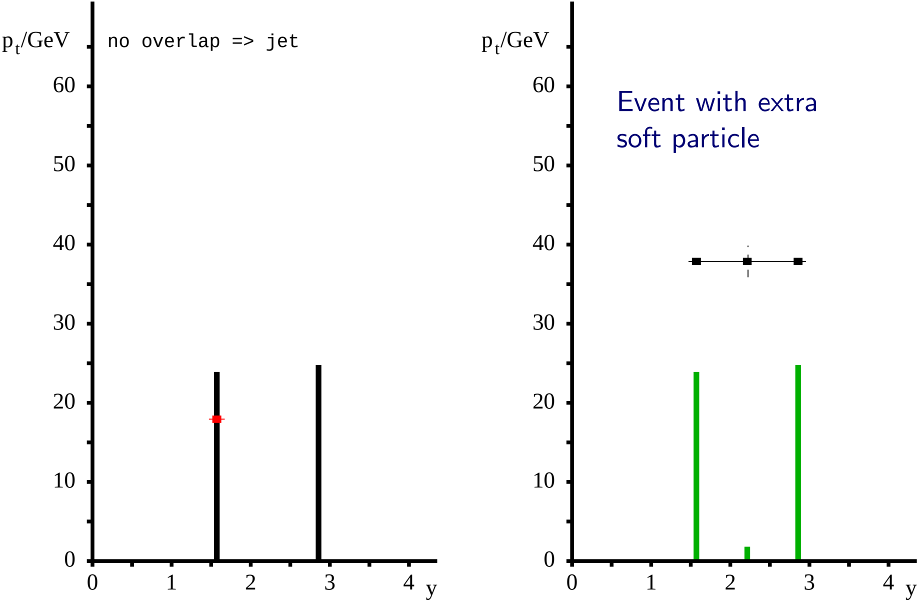


ICSM: infrared safety

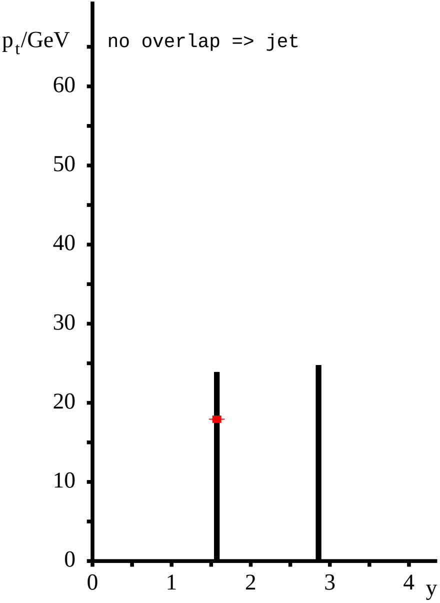


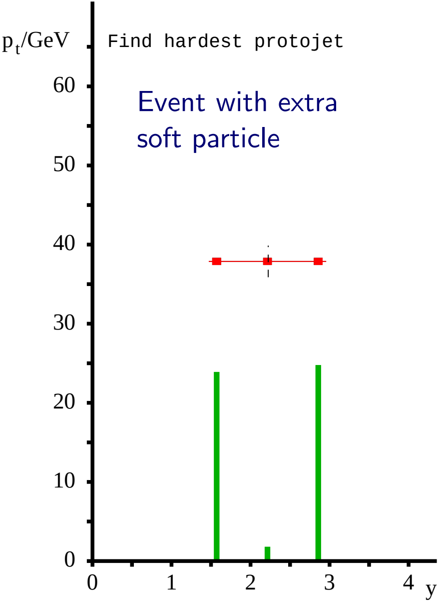


ICSM: infrared safety

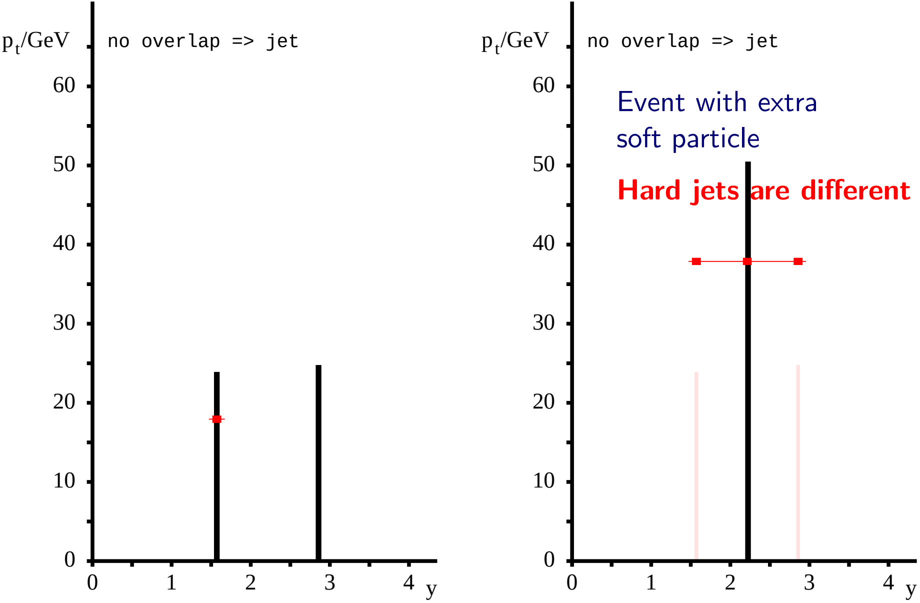


ICSM: infrared safety





ICSM: infrared safety



Lecture 1 $\longrightarrow$ 2

In lecture 1, we saw

- sequential recombination (k_t, etc.) algorithms
- the first of a series of cone-algorithms, those with "progressive removal" (xC-PR)
- and ran into collinear safety issues (from ordering of "seeds" for cone direction)

Today

- see the other series of cone-algorithms (with split–merge, xC-SM)
- look more at the physics of jet algs.

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

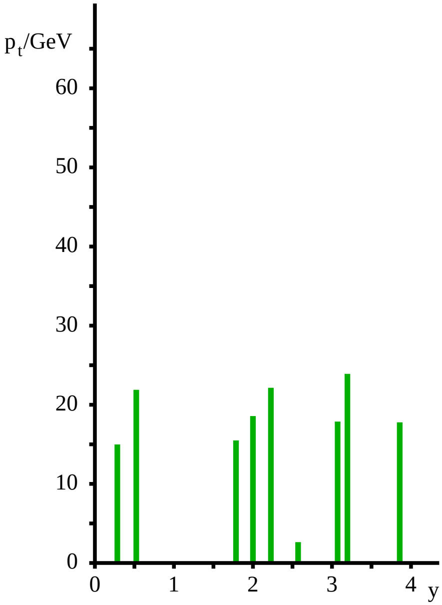
Unifying idea: momentum flow within a cone only marginally modified by QCD branching

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"

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

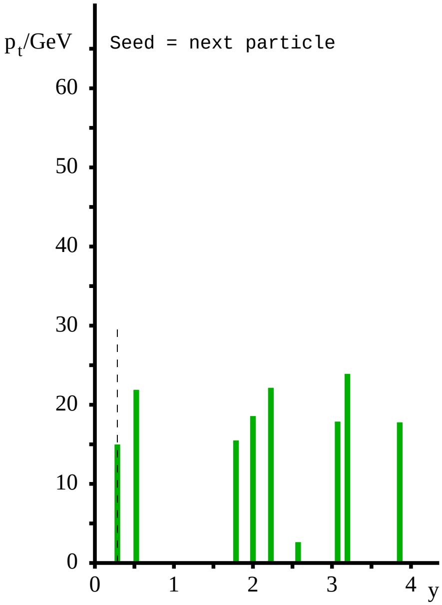
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

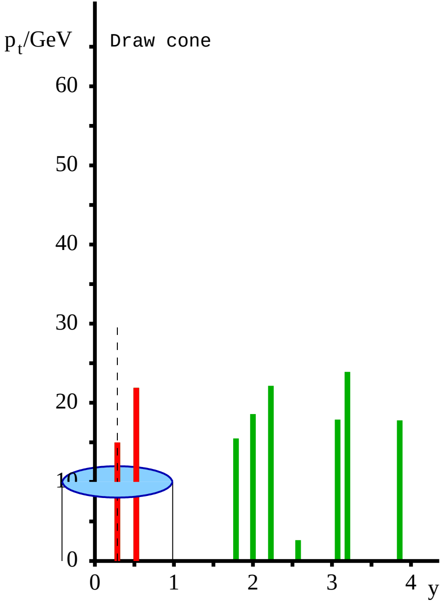
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

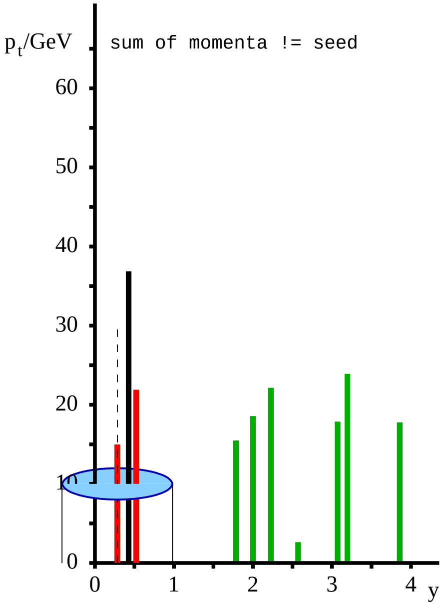
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

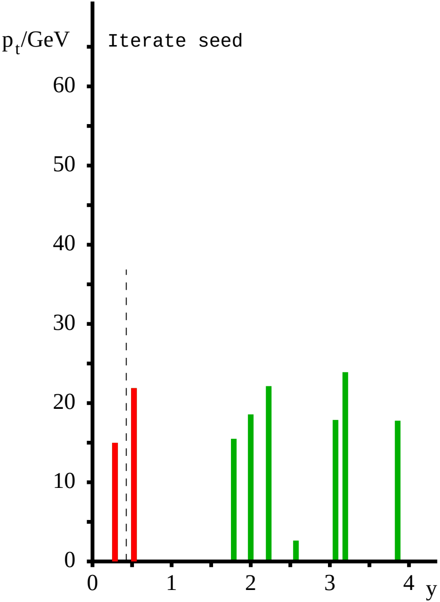
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

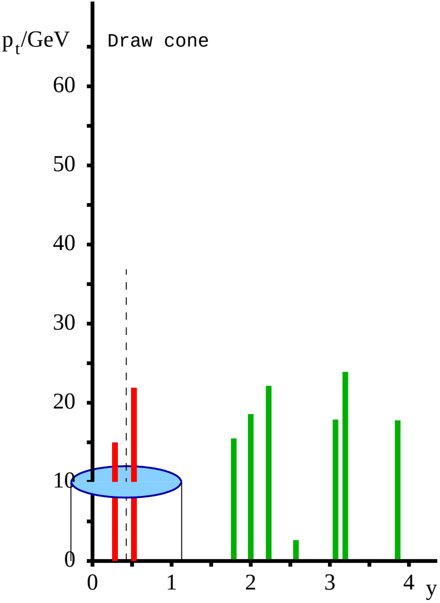
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

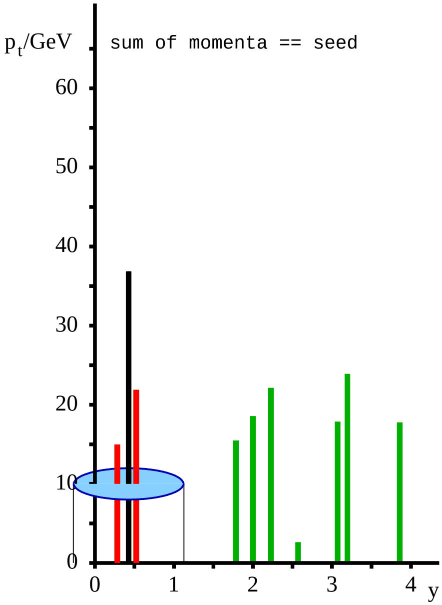
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

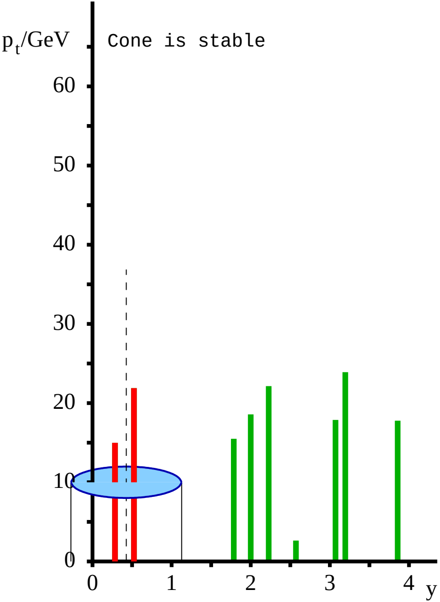
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

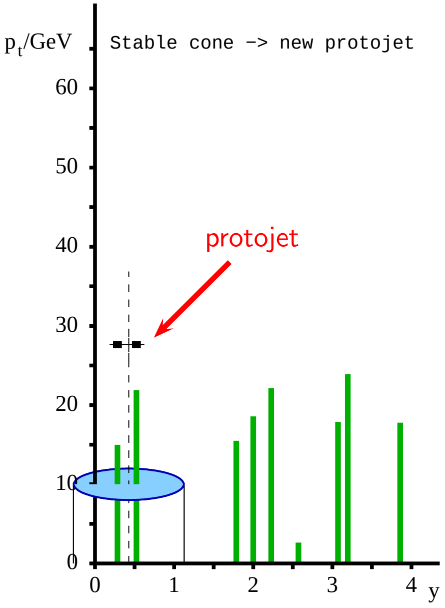
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



<u>Avoid ordering seeds (coll. unsafe)</u>

CDF JetClu† & ATLAS cones

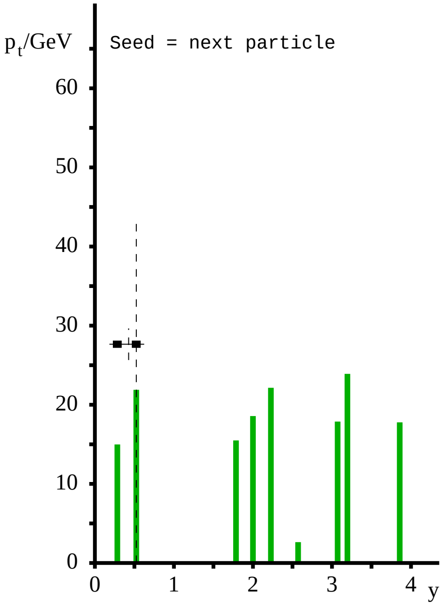
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

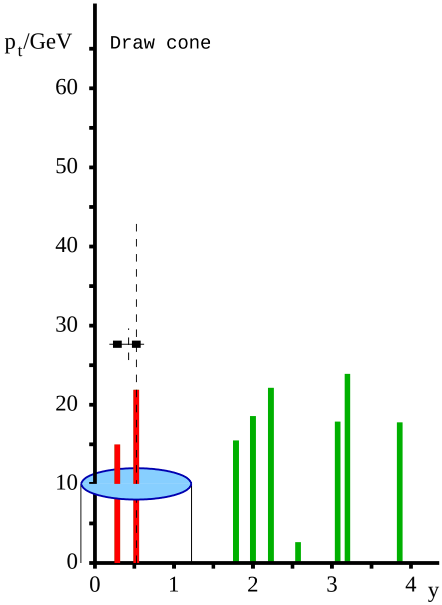
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

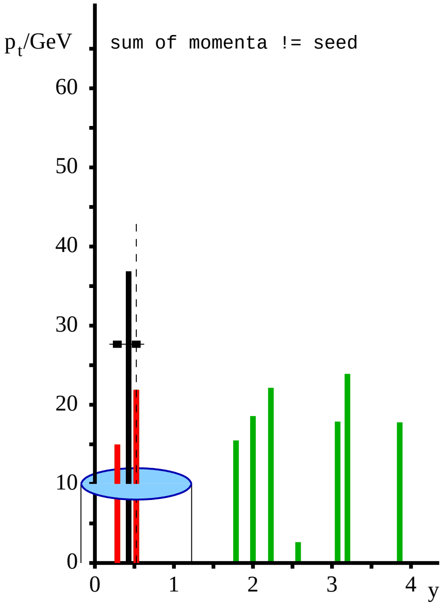
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

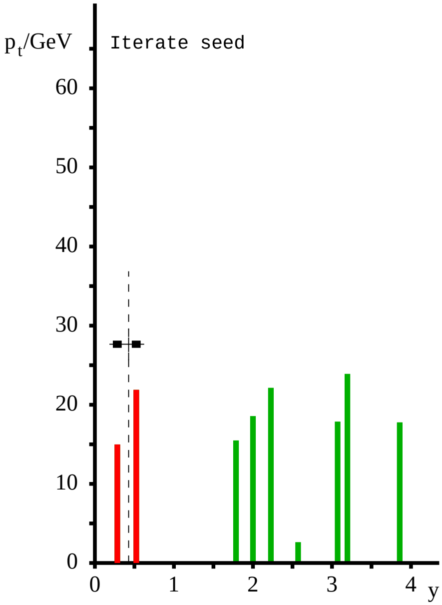
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

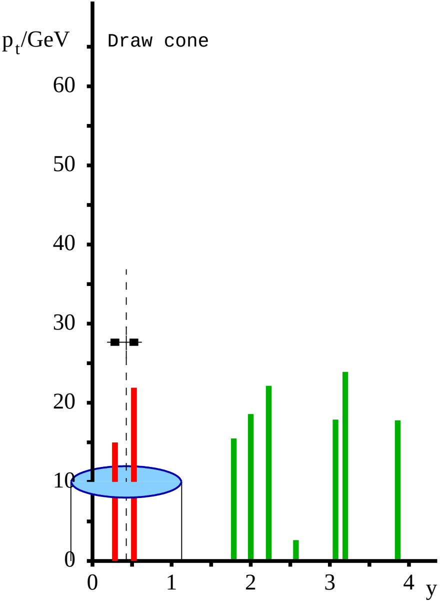
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

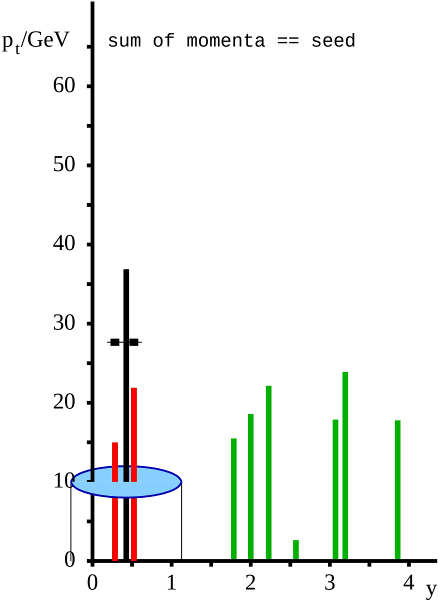
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

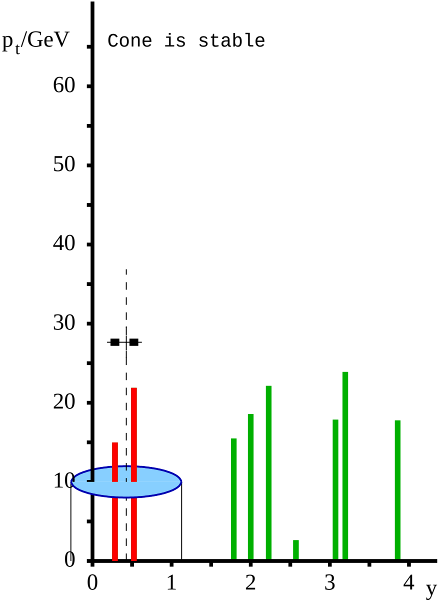
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

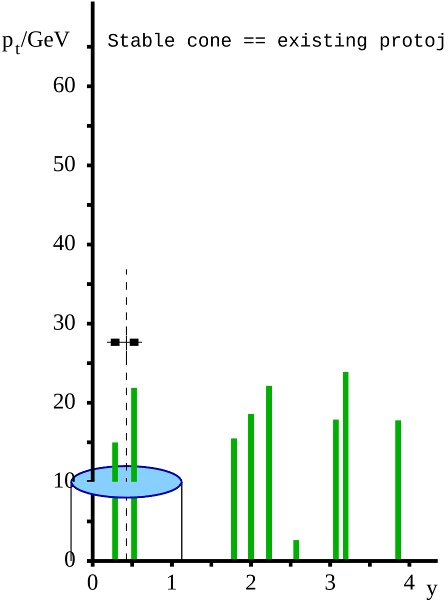
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

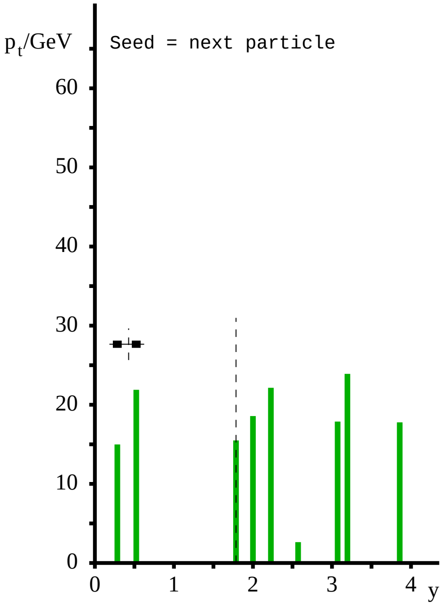
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

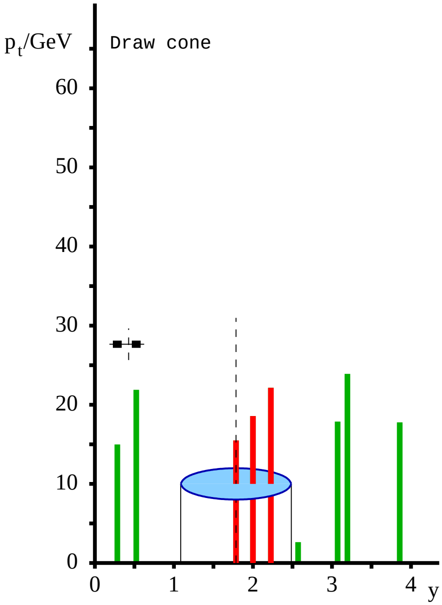
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

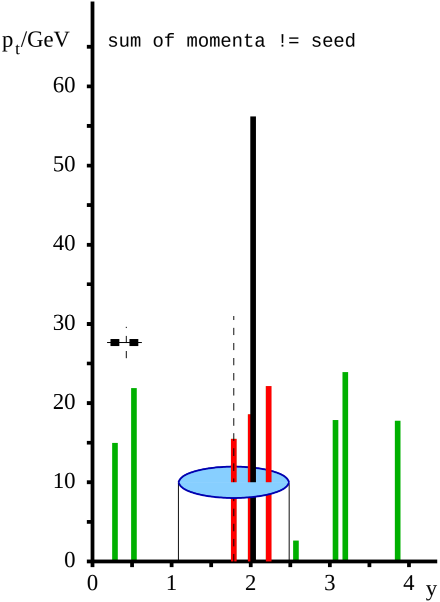
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



<u>Avoid ordering seeds (coll. unsafe)</u>

CDF JetClu† & ATLAS cones

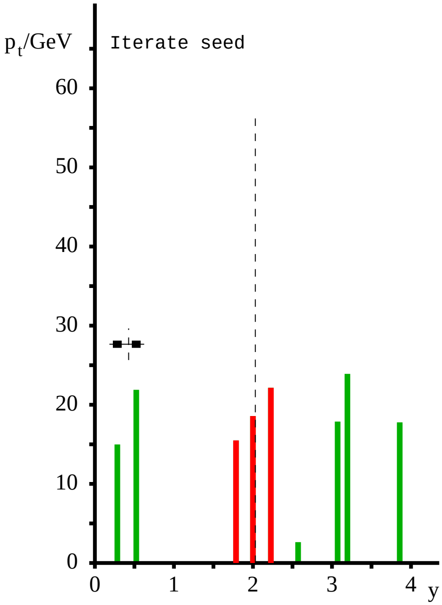
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

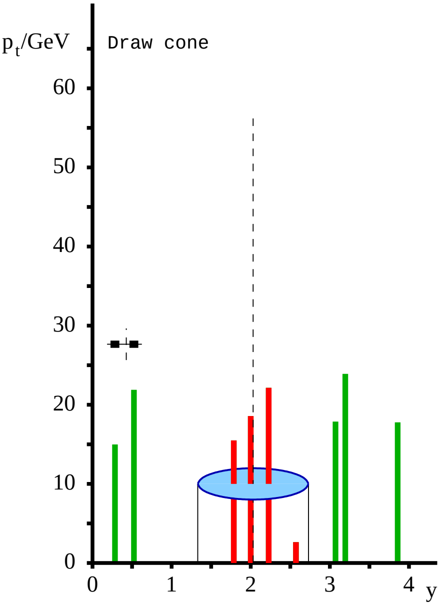
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

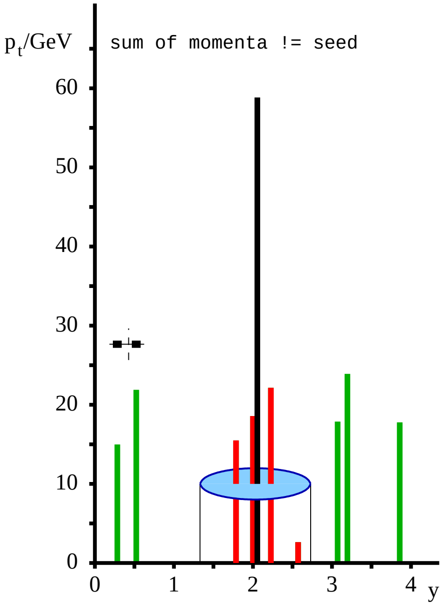
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

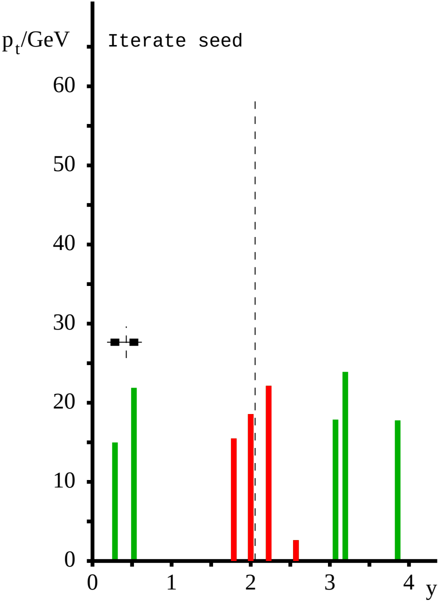
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

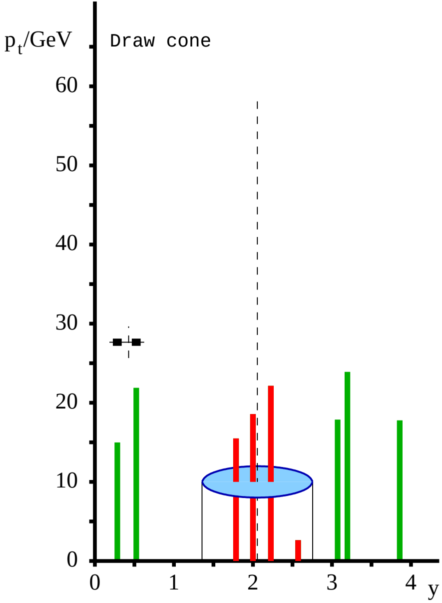
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

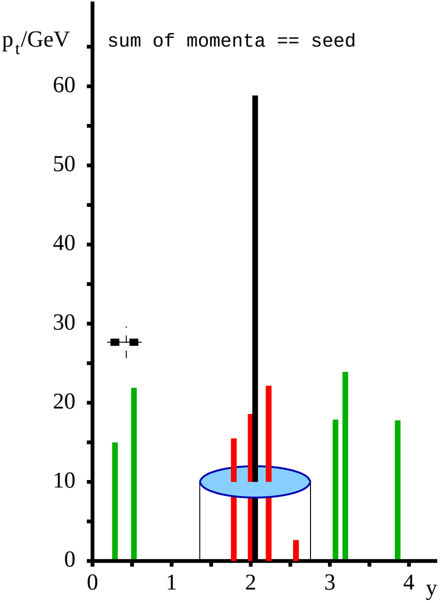
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

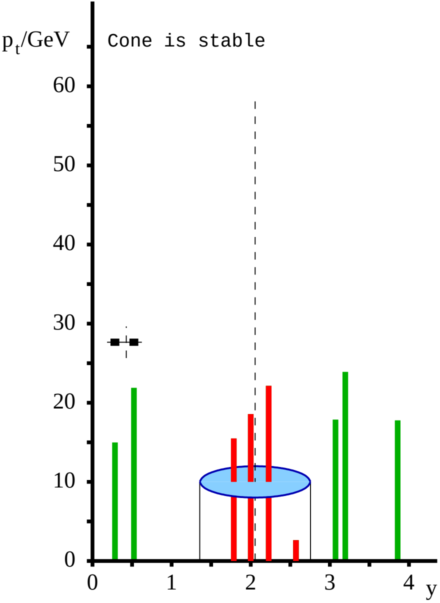
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets overlap. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a split–merge procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

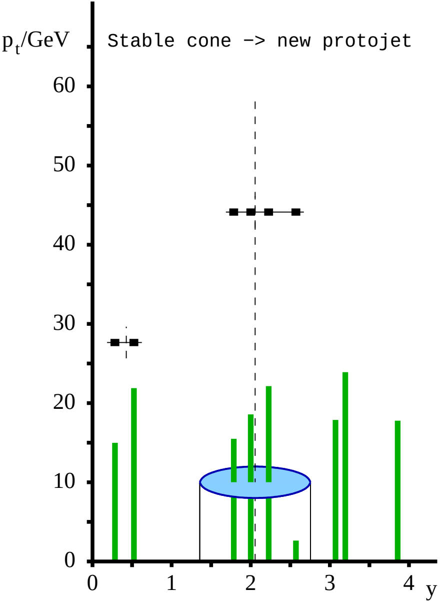
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet $\neq$ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

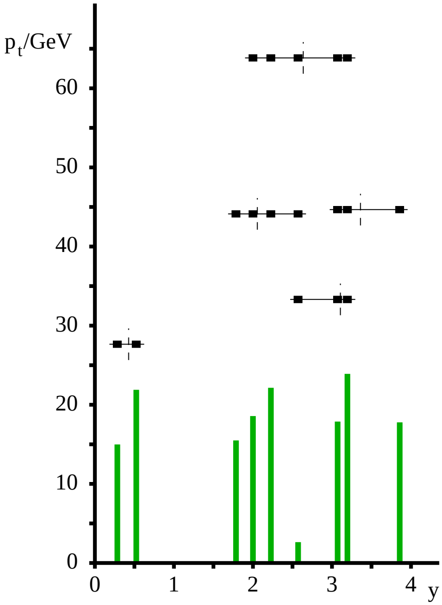
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

It. Cone with Split–Merge (IC-SM)



<u>Avoid ordering seeds (coll. unsafe)</u>

CDF JetClu† & ATLAS cones

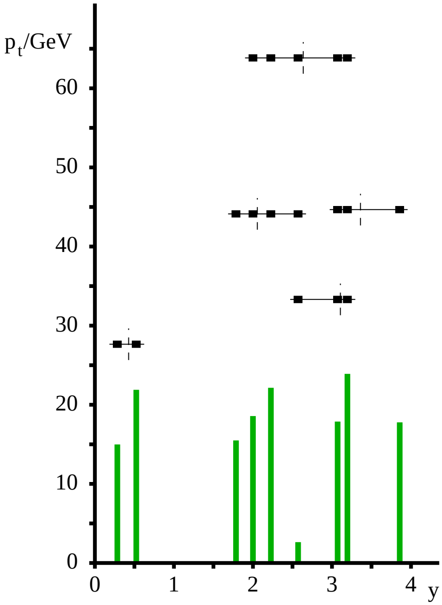
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

It. Cone with Split–Merge (IC-SM)



Avoid ordering seeds (coll. unsafe)

CDF JetClu† & ATLAS cones

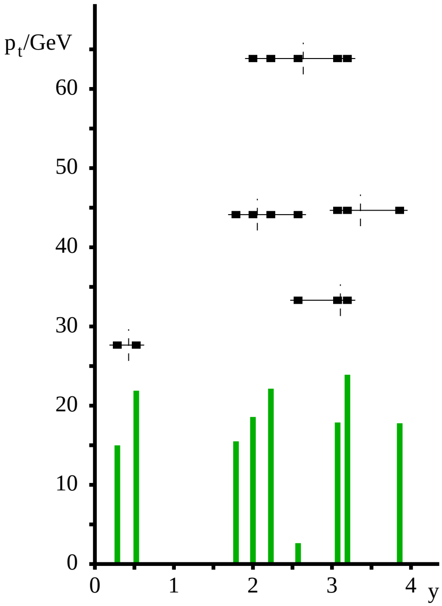
- use every particle as possible seed (no particular order)
- iterate until stable cone
- add the stable cone to the list of **protojets** unless it's already there
- until all seeds done

Note: protojets **overlap**. Certain particles appear in many protojets

protojet ≠ jet

Must resolve the overlaps.
Use a **split–merge** procedure.

IC-SM: split–merge part



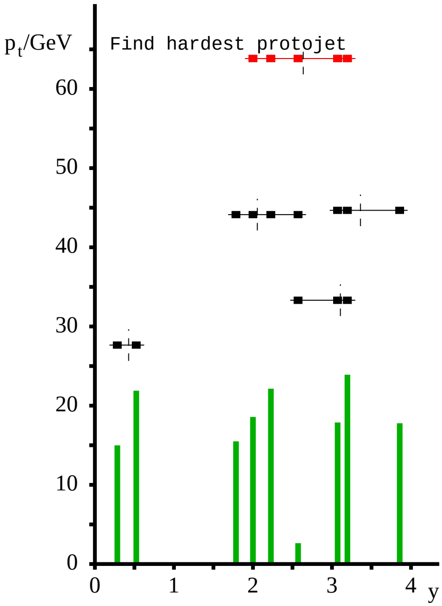
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\to$ jet.
- repeat...

IC-SM: split–merge part



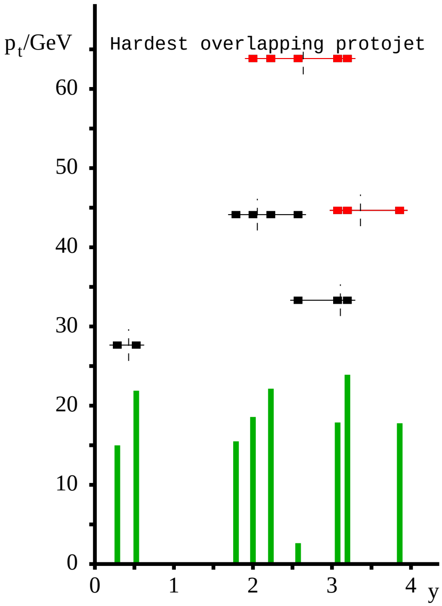
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



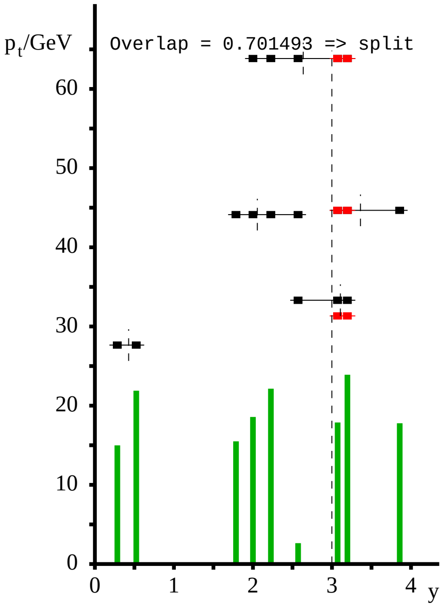
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



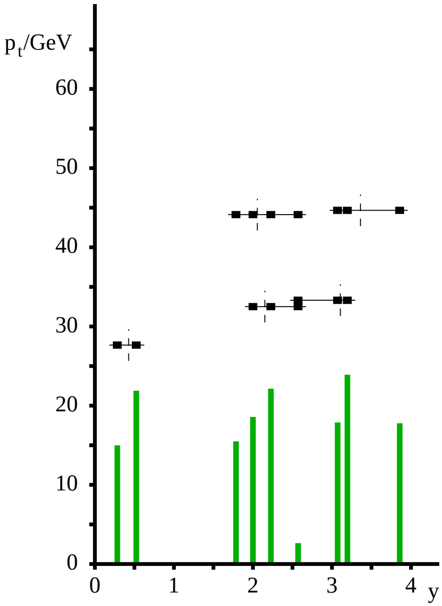
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



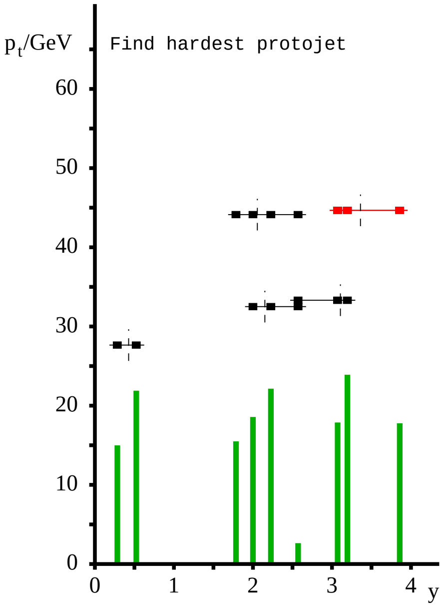
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



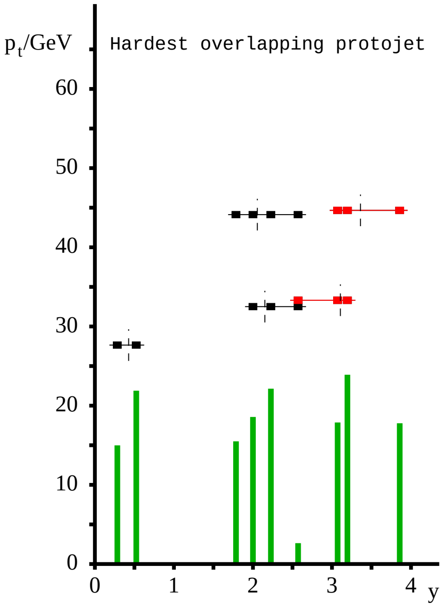
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared} / p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\to$ jet.
- repeat...

IC-SM: split–merge part



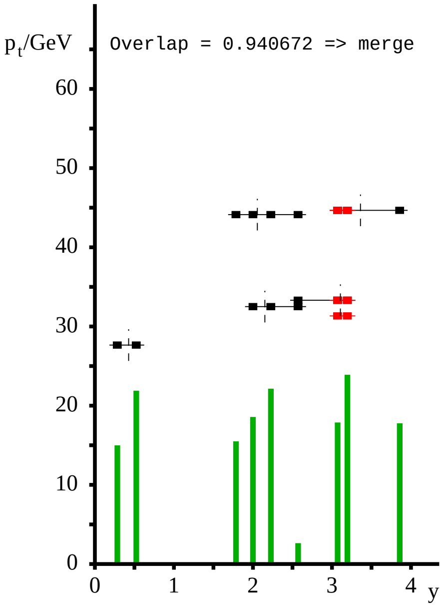
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared} / p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\to$ jet.
- repeat. . .

IC-SM: split–merge part



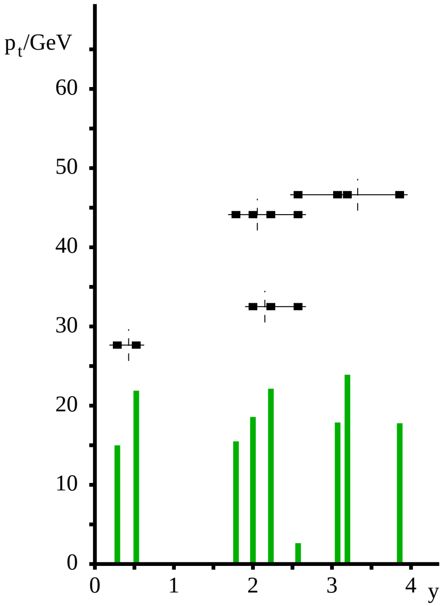
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



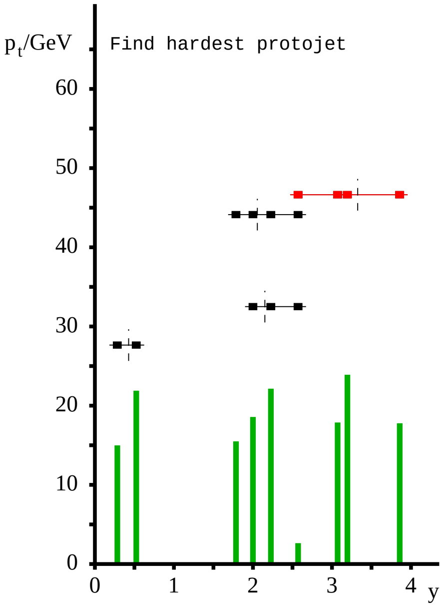
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



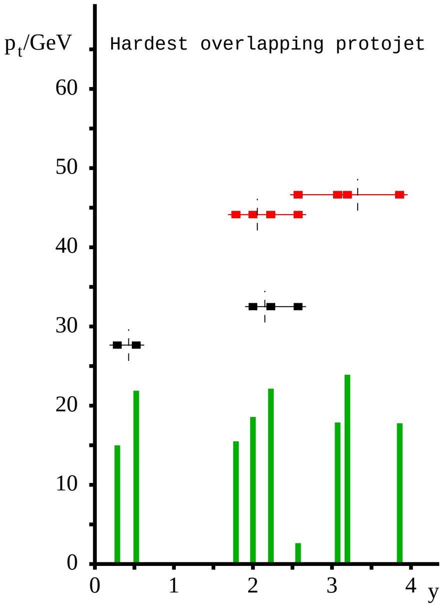
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ → jet.
- repeat...

IC-SM: split–merge part



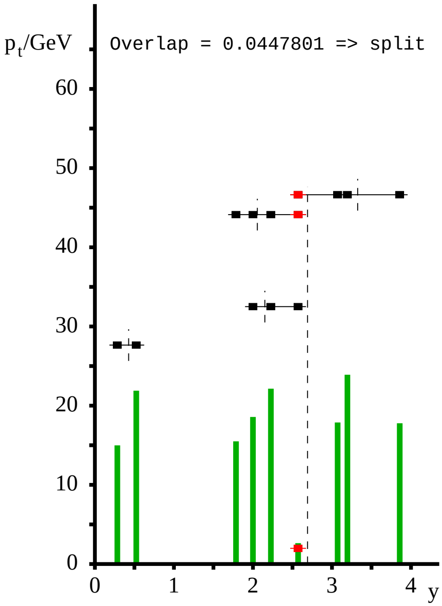
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



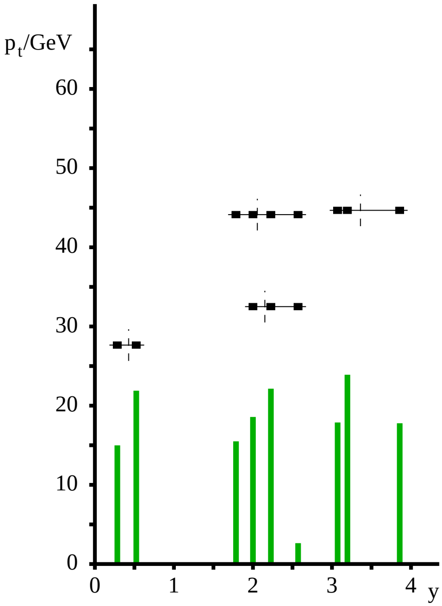
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



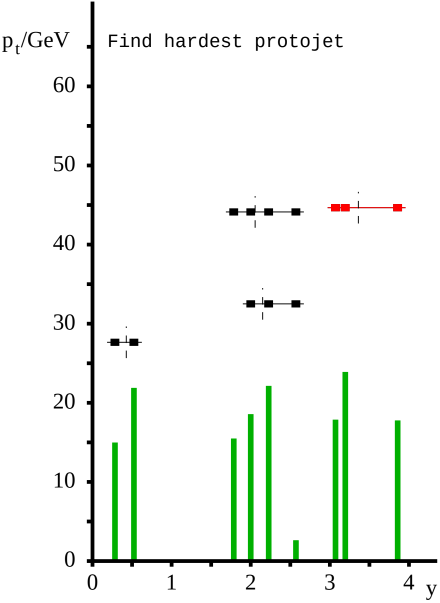
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



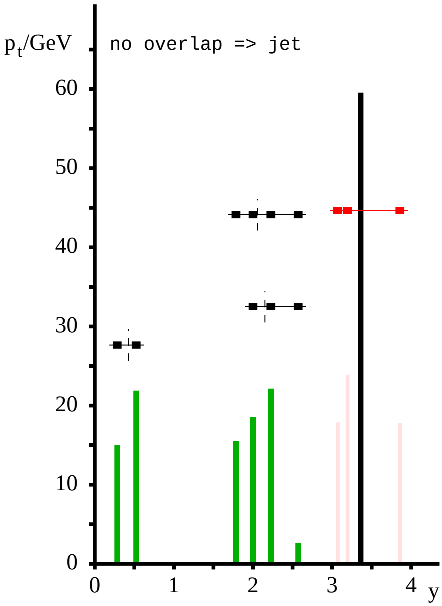
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared} / p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



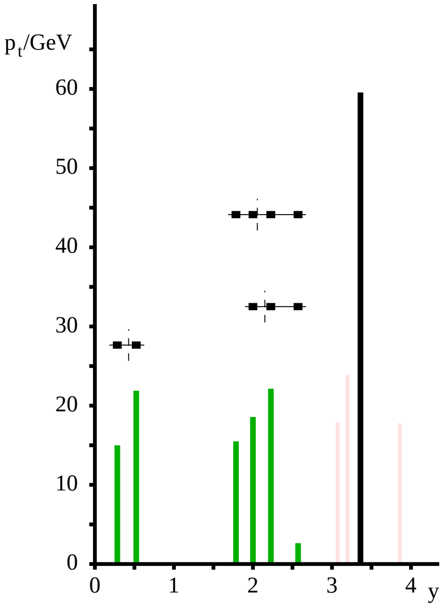
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



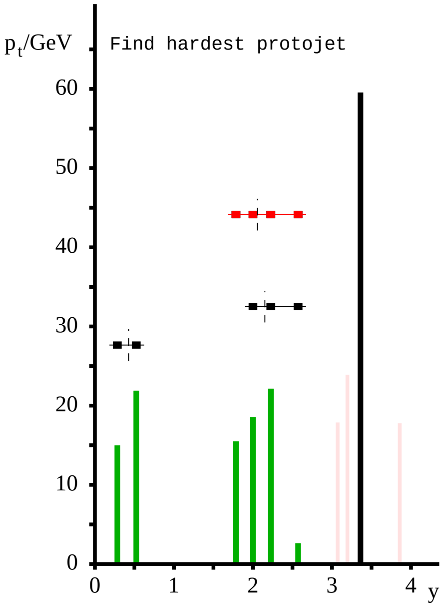
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



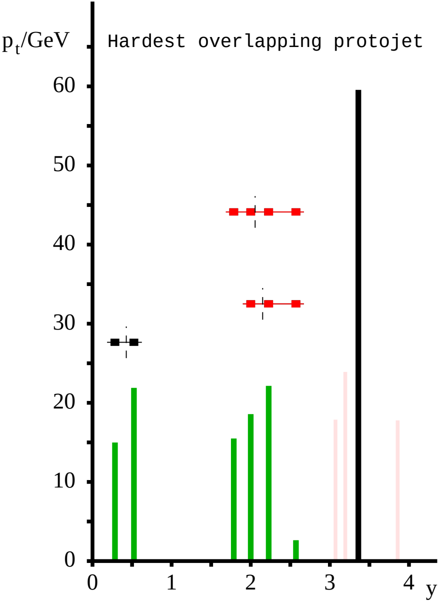
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\to$ jet.
- repeat. . .

IC-SM: split–merge part



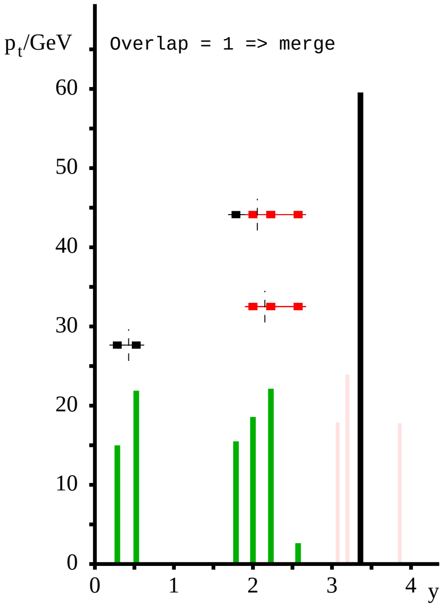
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared} / p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat...

IC-SM: split–merge part



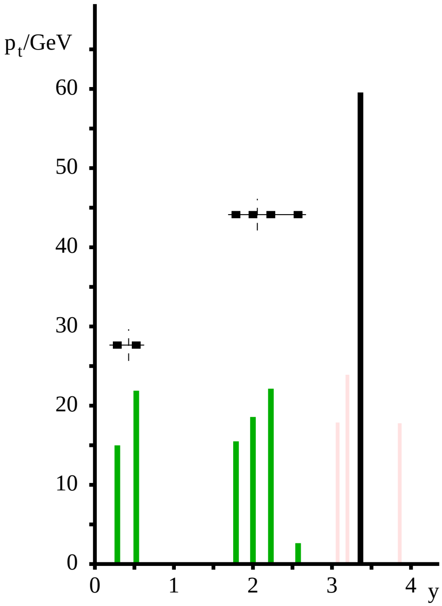
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



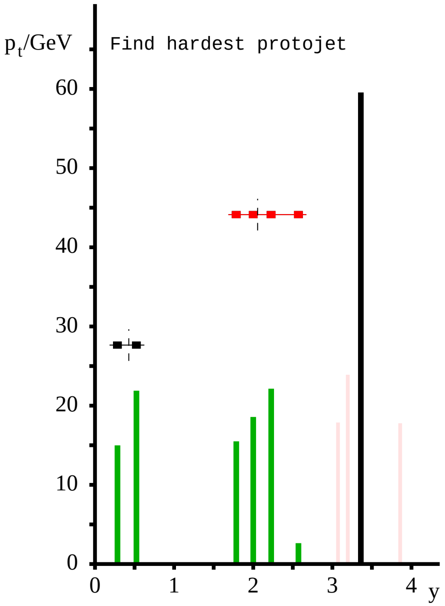
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



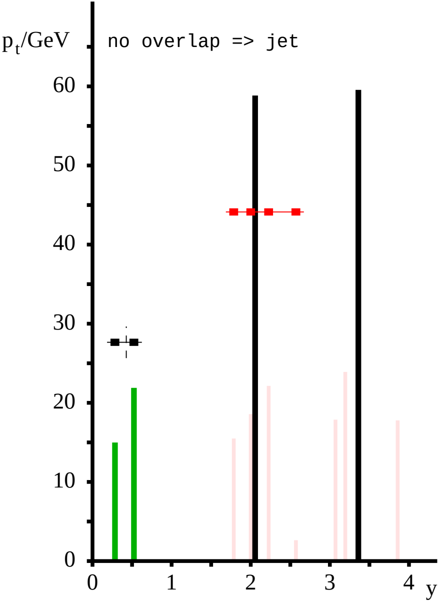
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



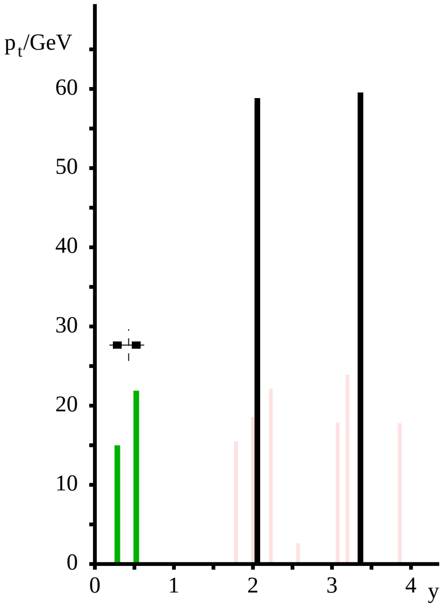
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ → jet.
- repeat. . .

IC-SM: split–merge part



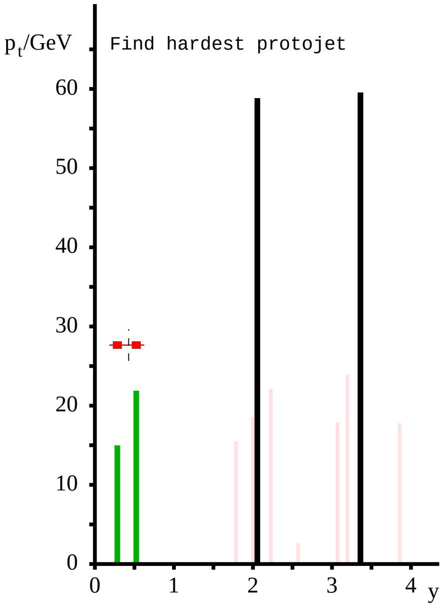
SM in Tevatron Run II formulation

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ → jet.
- repeat. . .

IC-SM: split–merge part



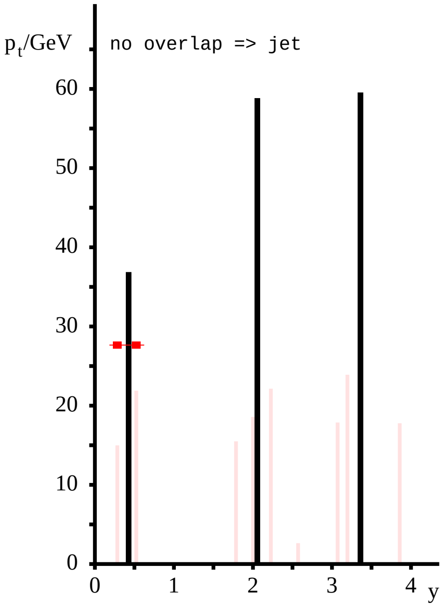
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



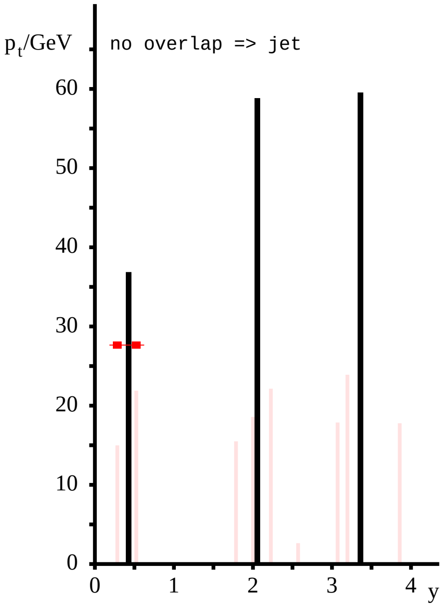
<u>SM in Tevatron Run II formulation</u>

but common to most xC-SM

Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

IC-SM: split–merge part



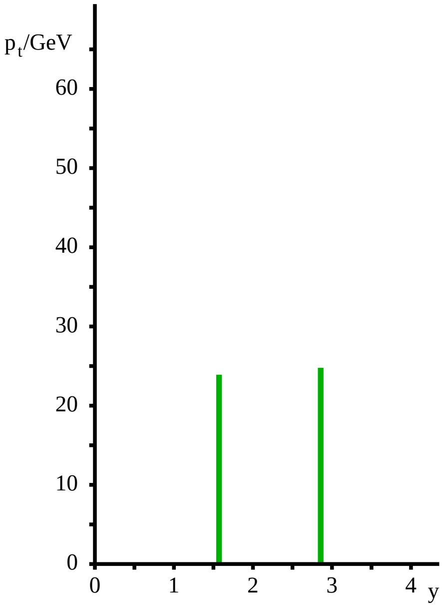
SM in Tevatron Run II formulation

but common to most xC-SM

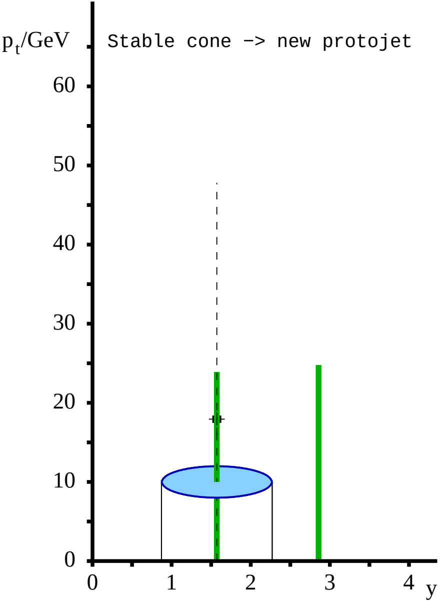
Introduce overlap threshold f

- Identify hardest protojet (PJ), p_1
- Find hardest PJ that overlaps with it, p_2
- Calculated overlap, $O = p_{t,shared}/p_{t,2}$
  - if $O < f$, split along axis at center of two PJs
  - if $O > f$ merge the two PJs
- If there is no overlap, PJ $\rightarrow$ jet.
- repeat. . .

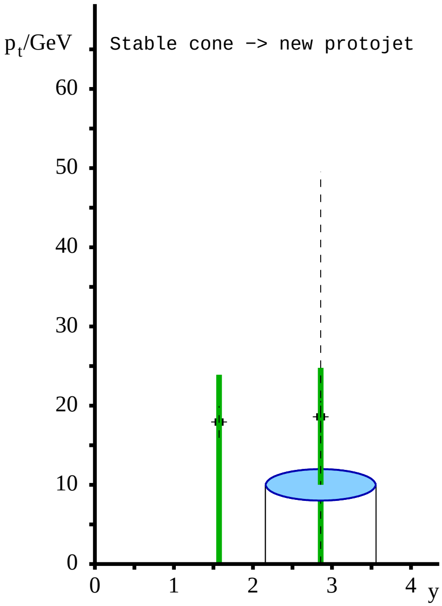
IC-SM: infrared safety



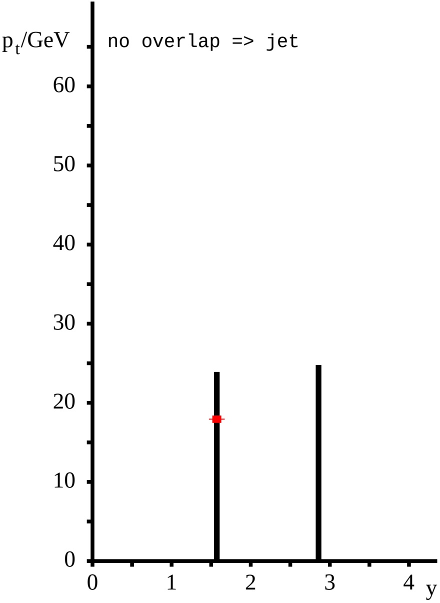
IC-SM: infrared safety



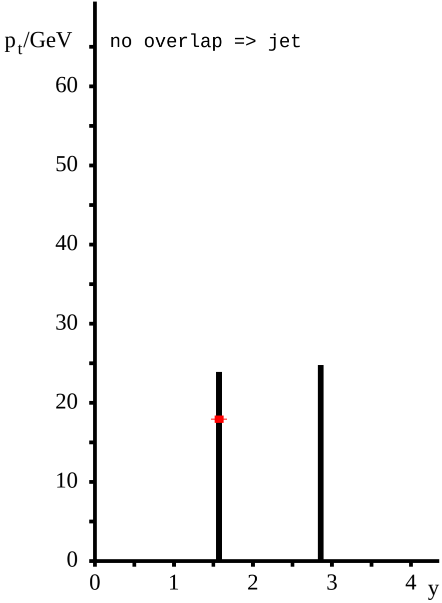
IC-SM: infrared safety

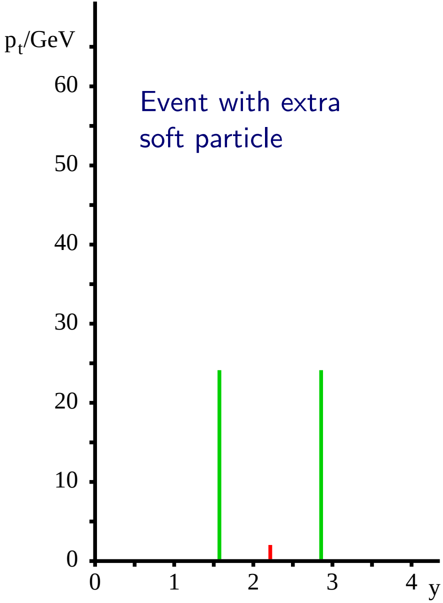


IC-SM: infrared safety

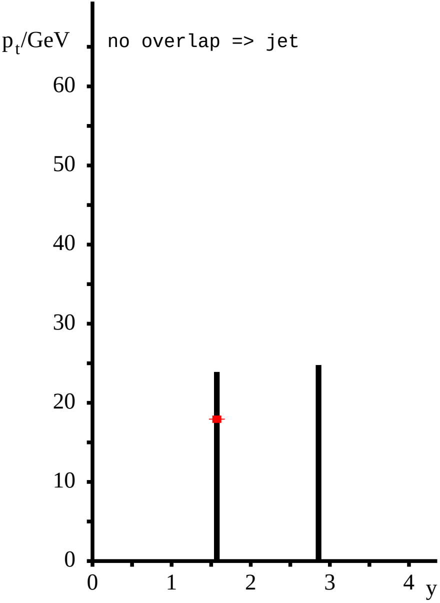


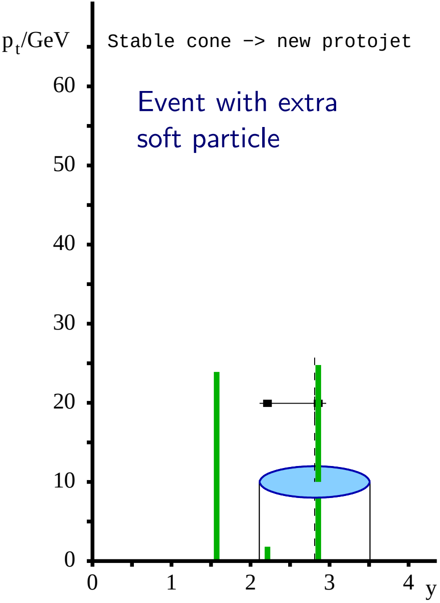
IC-SM: infrared safety



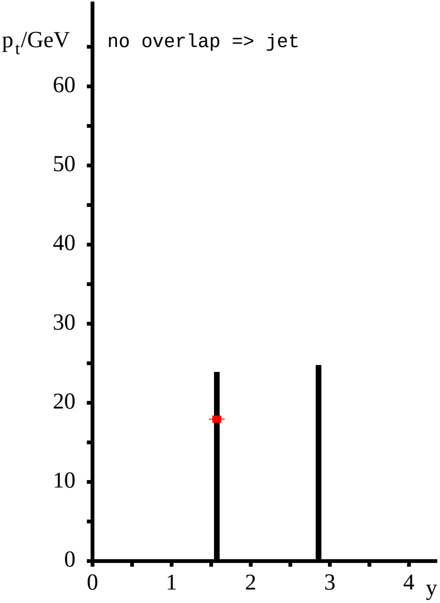


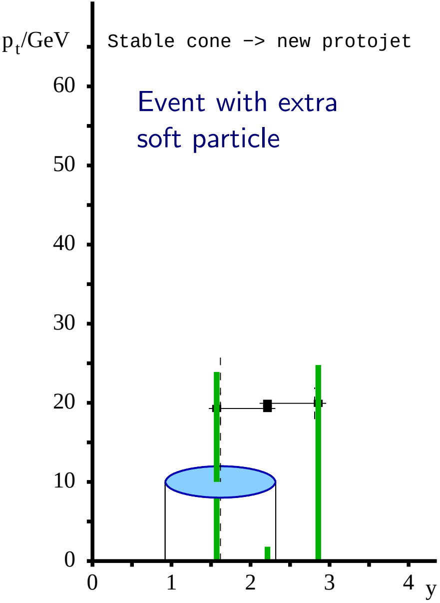
IC-SM: infrared safety



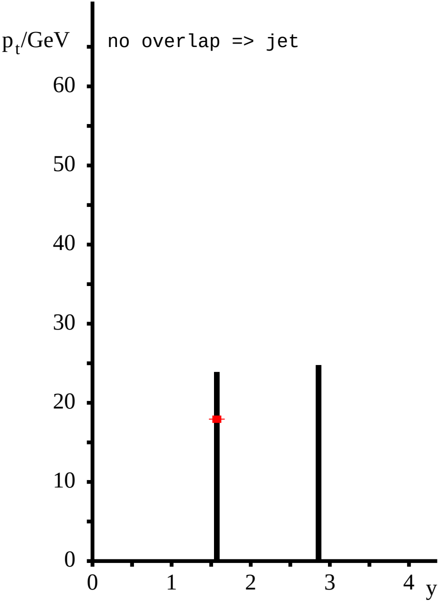


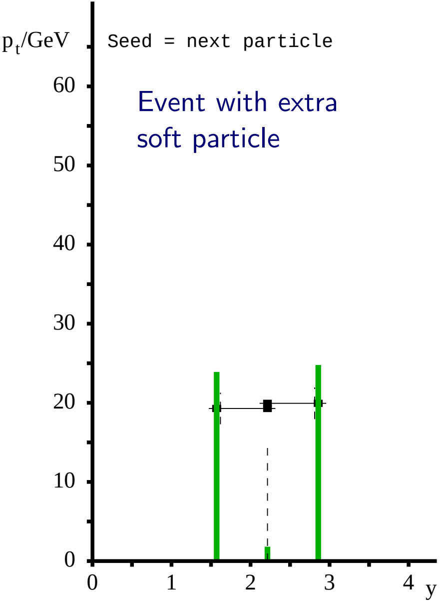
IC-SM: infrared safety



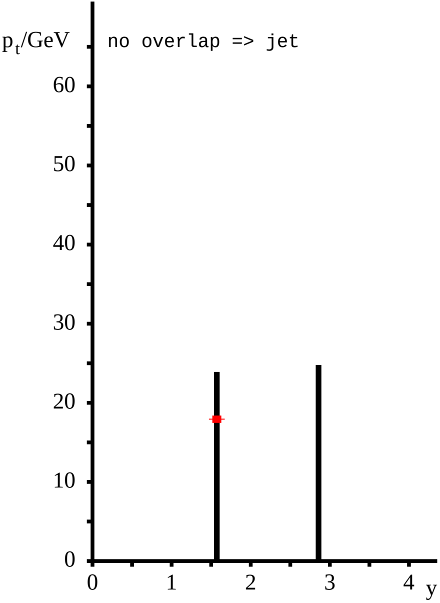


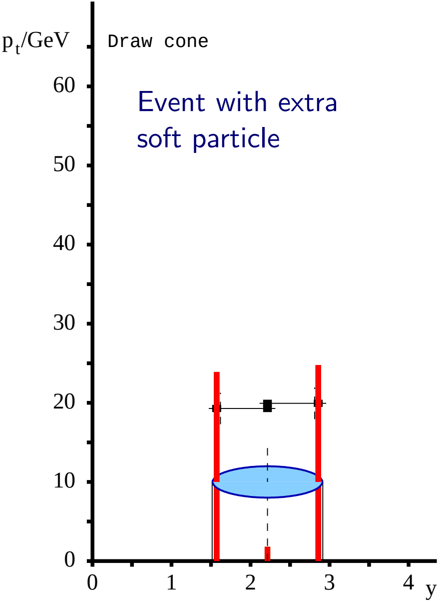
IC-SM: infrared safety





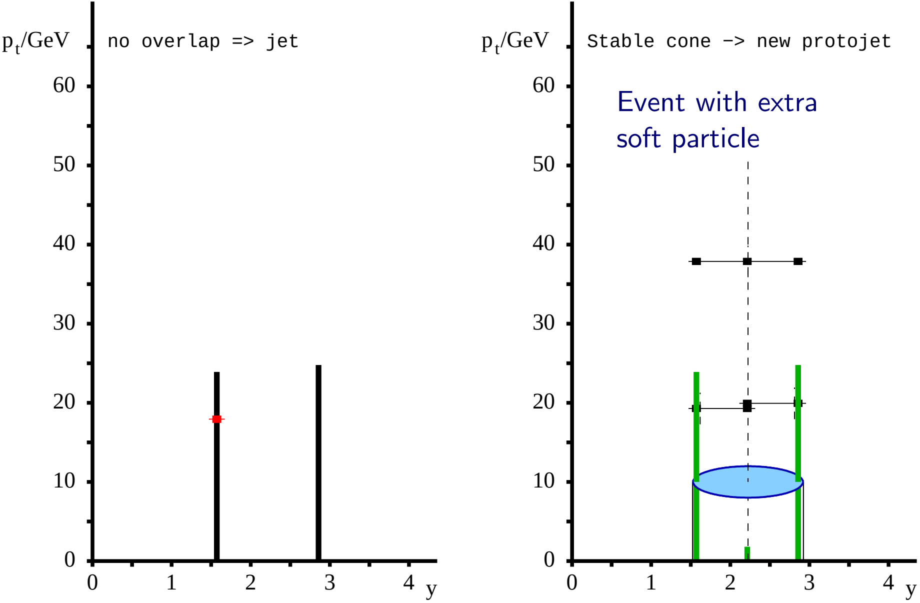
IC-SM: infrared safety



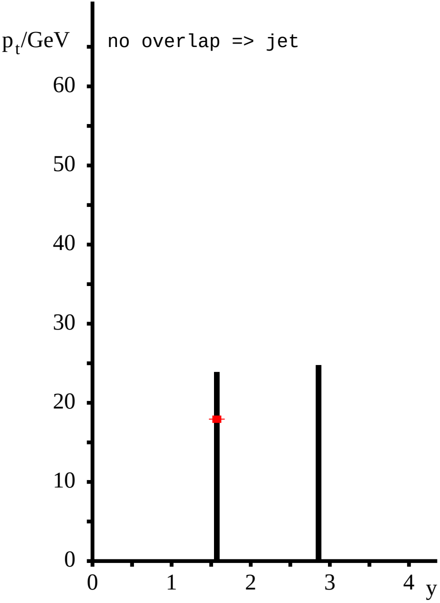


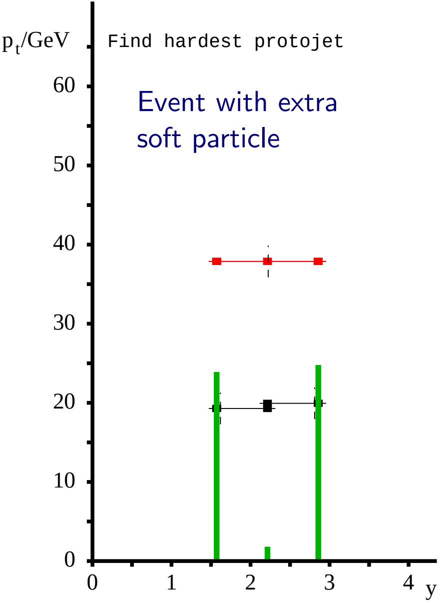
Event with extra soft particle

IC-SM: infrared safety

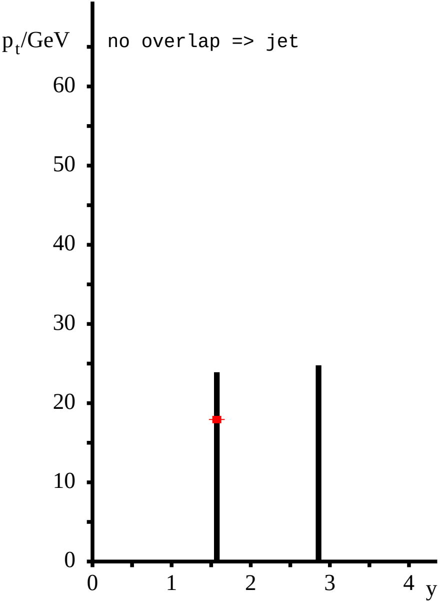


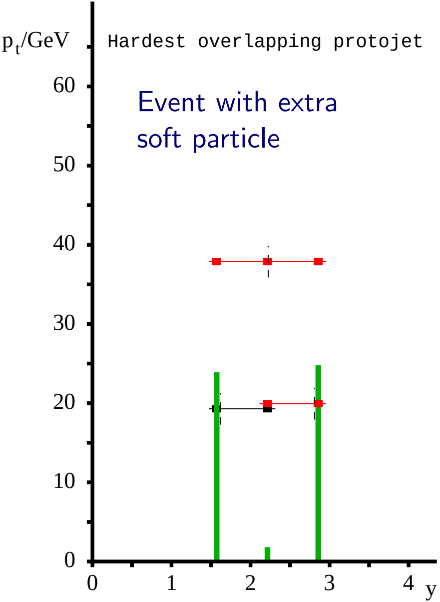
IC-SM: infrared safety



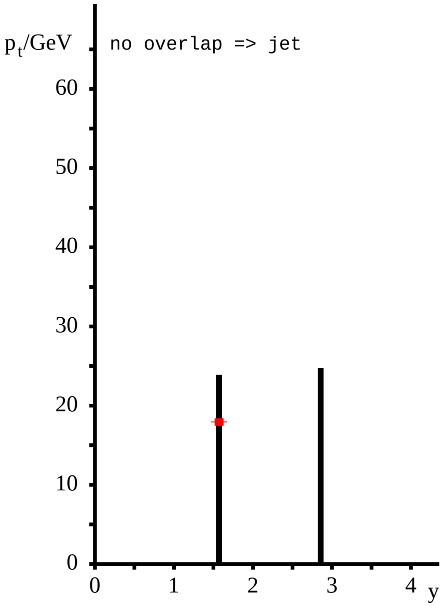


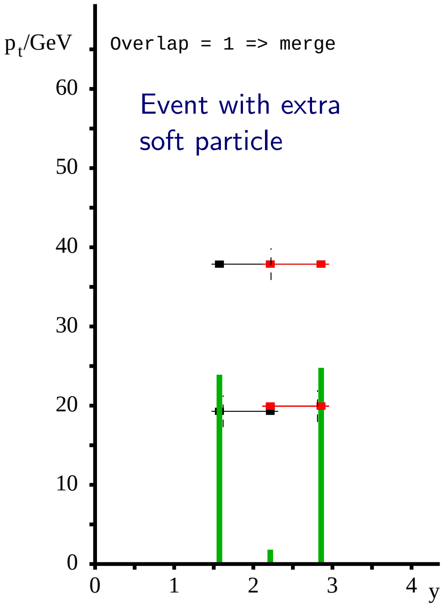
IC-SM: infrared safety



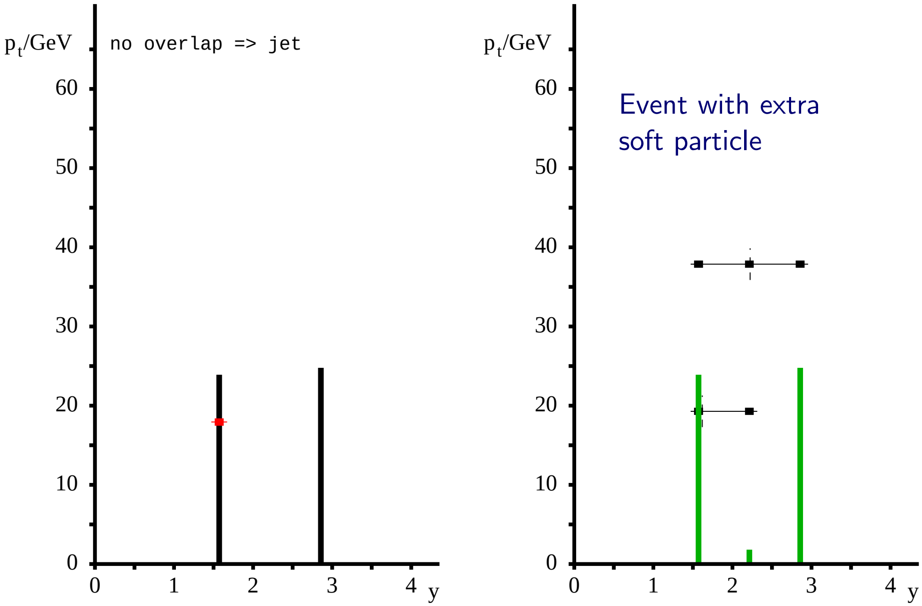


IC-SM: infrared safety

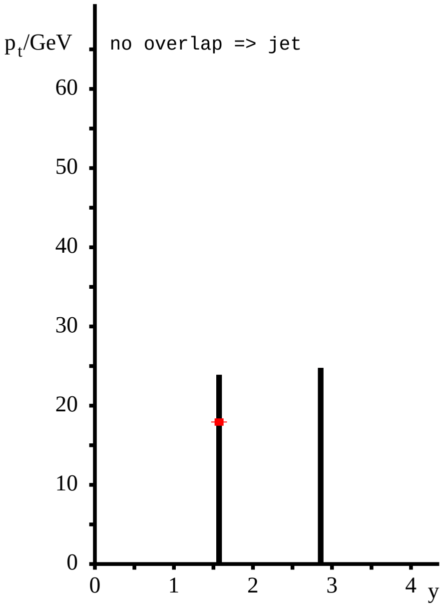


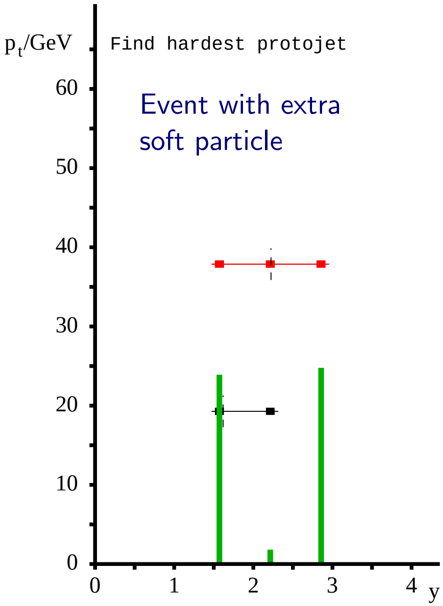


IC-SM: infrared safety



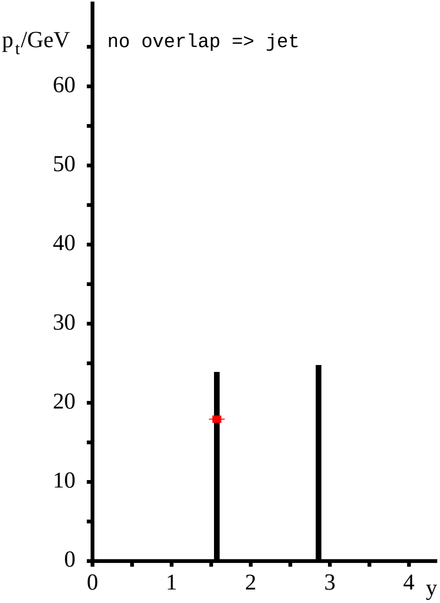
IC-SM: infrared safety

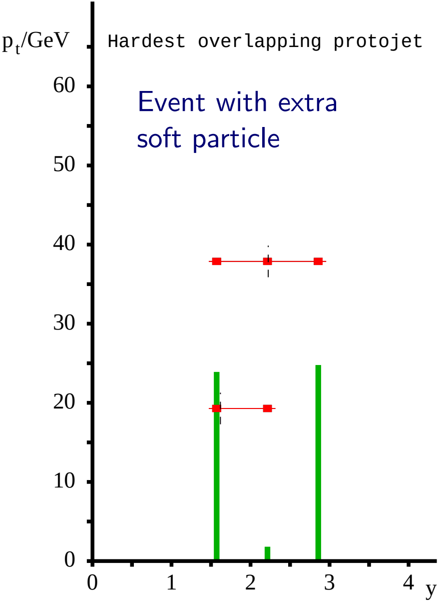




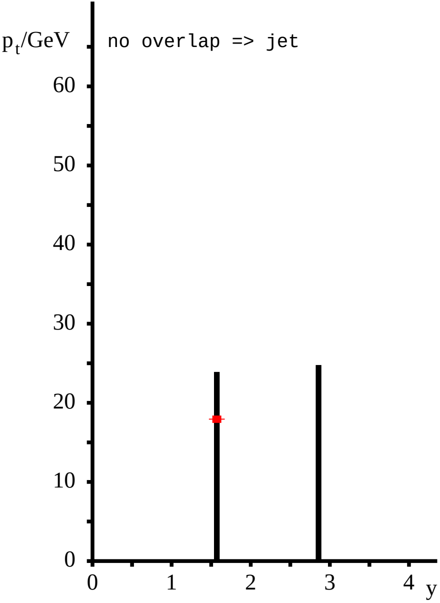
Event with extra soft particle

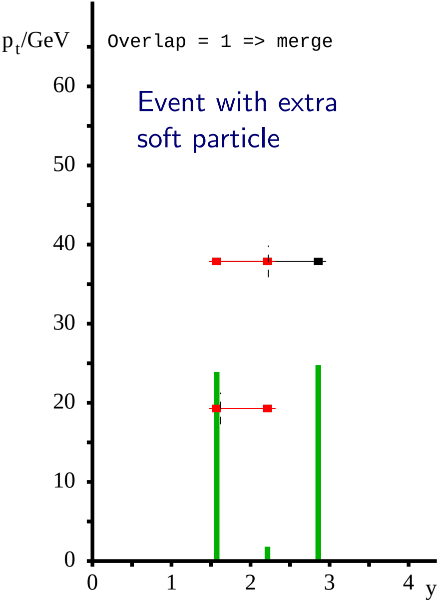
IC-SM: infrared safety



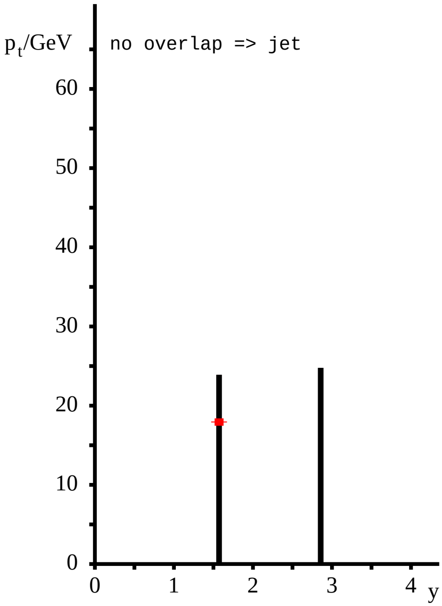


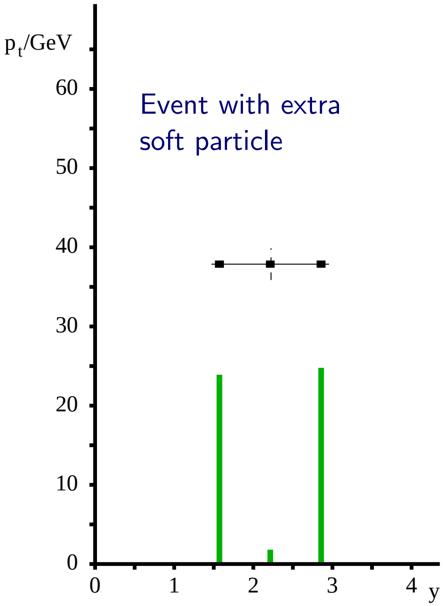
IC-SM: infrared safety



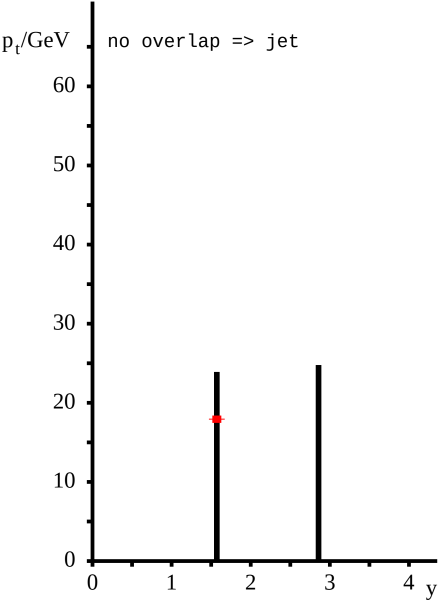


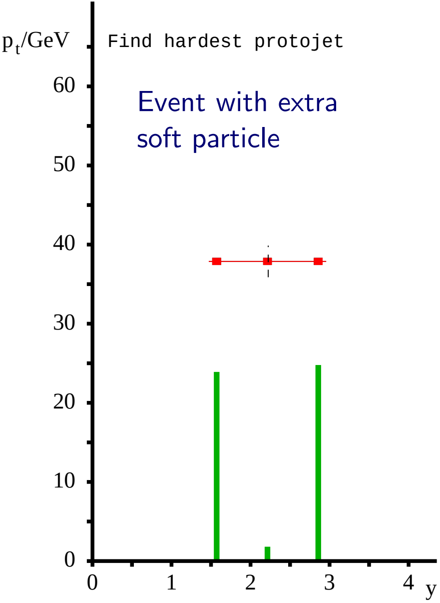
IC-SM: infrared safety





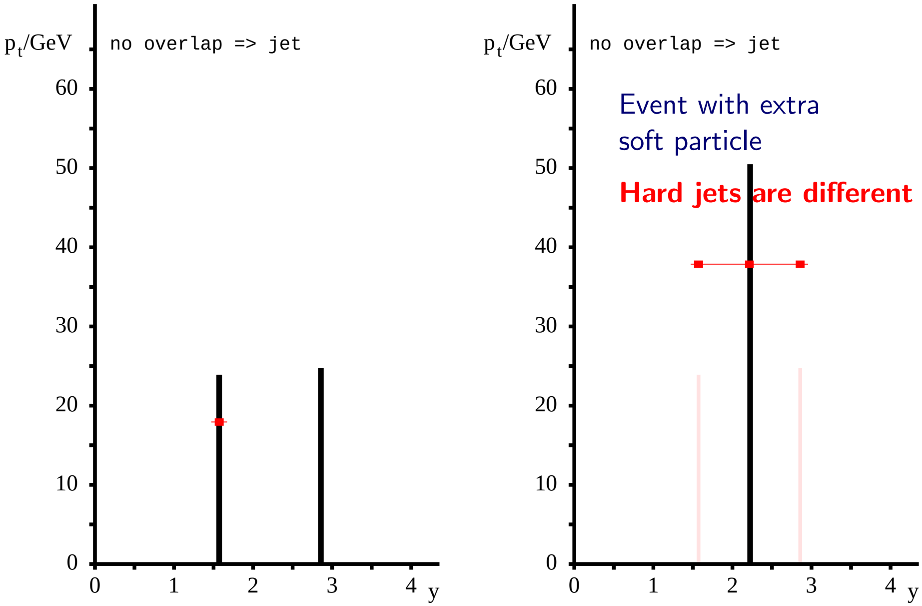
IC-SM: infrared safety





Event with extra soft particle

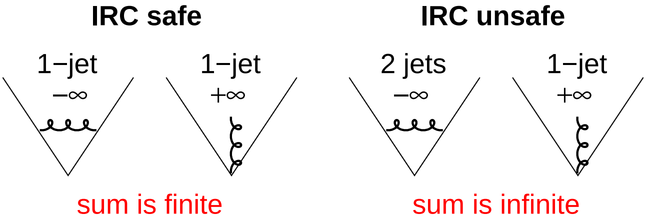
IC-SM: infrared safety



IRC safety crucial for theory

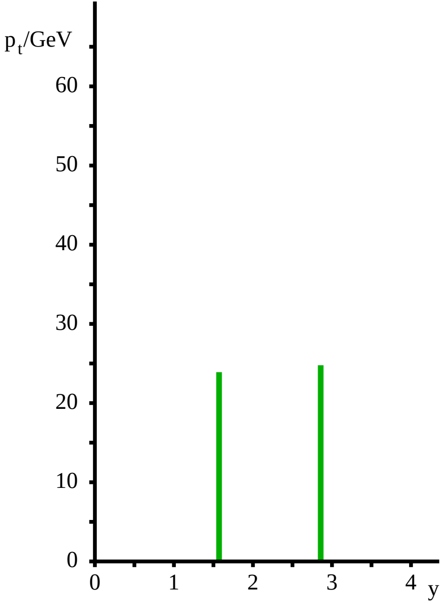
Soft emission, collinear splitting are both **infinite** in pert. QCD.

Infinities **cancel** with loop diagrams if jet-alg IRC safe



Some calculations simply become **meaningless**

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

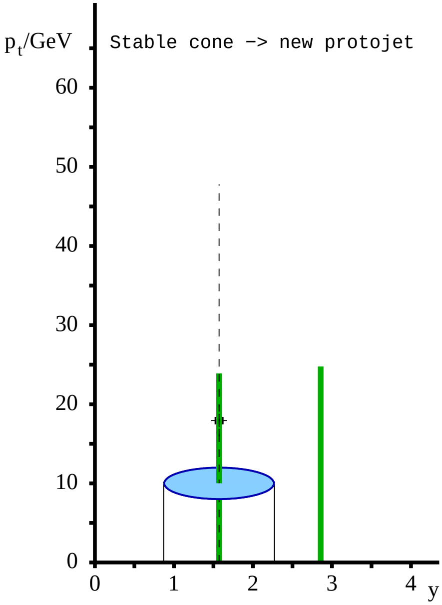
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

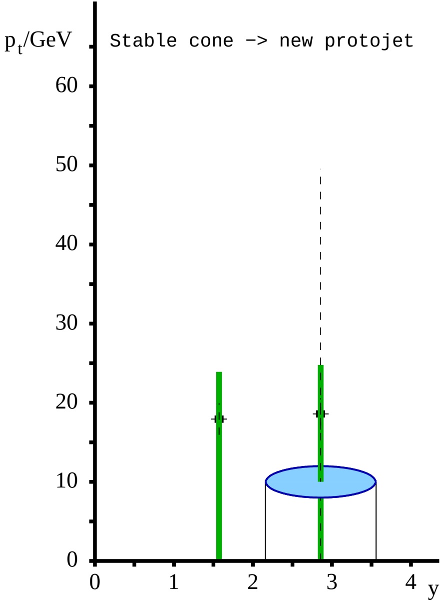
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC$_{mp}$-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

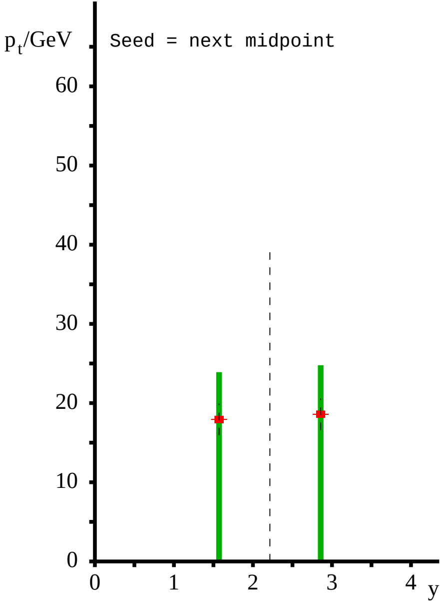
Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

CDF Midpoint algorithm
D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC$_{mp}$-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

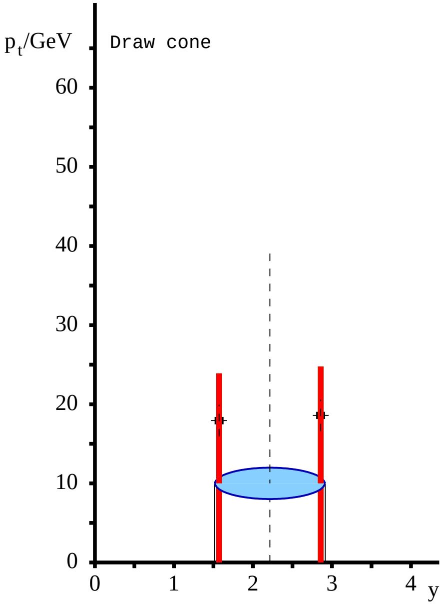
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

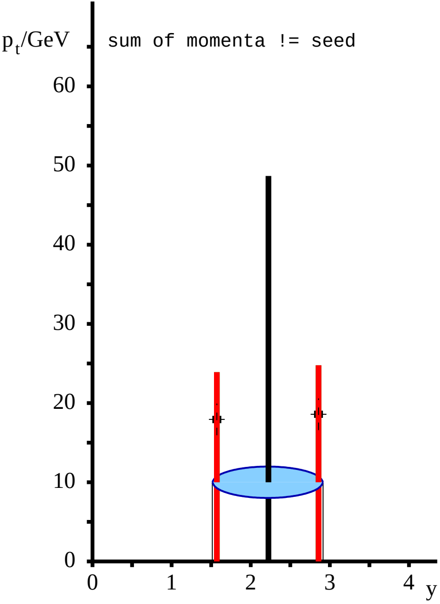
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

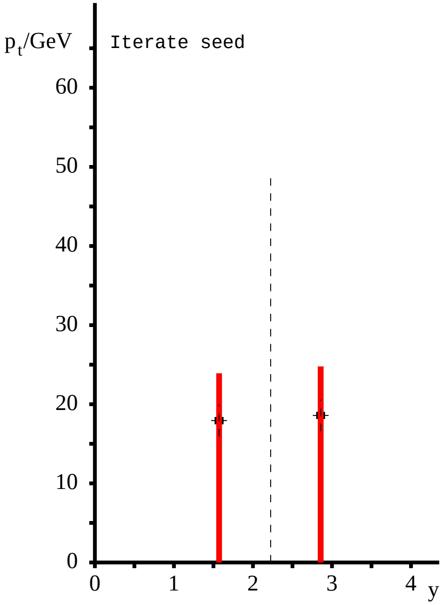
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

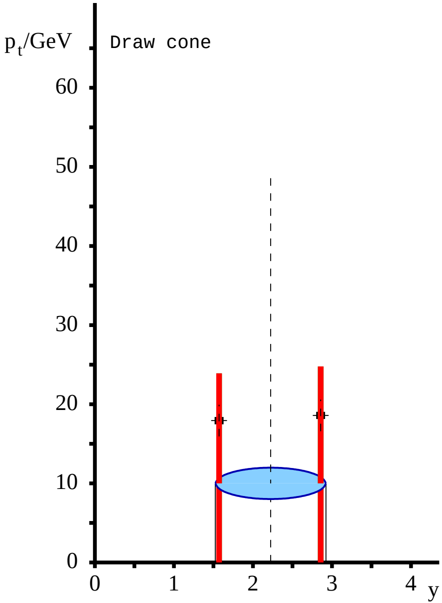
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

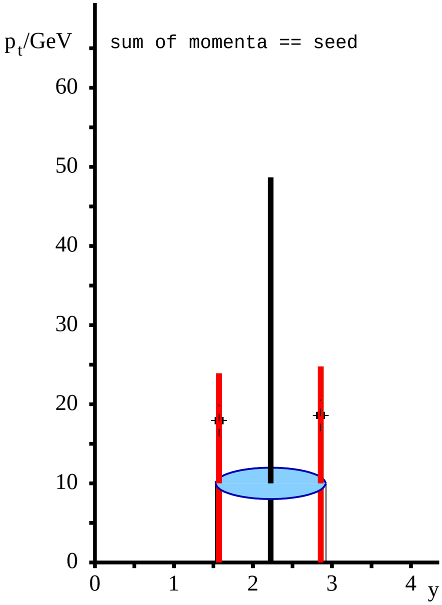
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC$_{mp}$-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

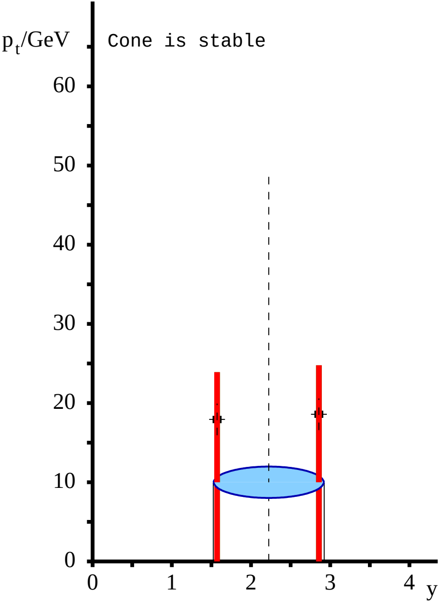
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC_{mp}-SM)



Looking for stable cones ≃ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

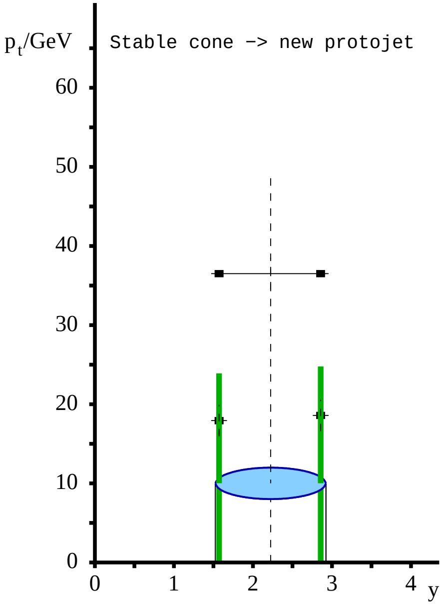
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC$_{mp}$-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

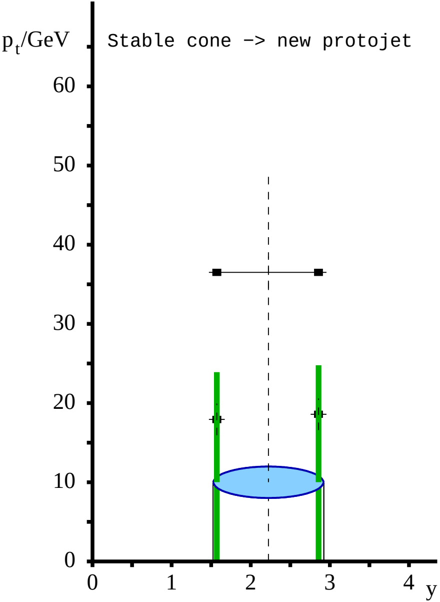
Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

CDF Midpoint algorithm
D0 Run II algorithm

This solves problem for
2-hard-particle configs.

[But it persists for 3-hard]

Midpoint algorithm (IC$_{mp}$-SM)



Looking for stable cones $\simeq$ finding local minima of a potential.

Problem: set of iterative solution depends on set of starting points.

Patch: after 1st round of iteration, find midpoints between protojets, use as new seeds

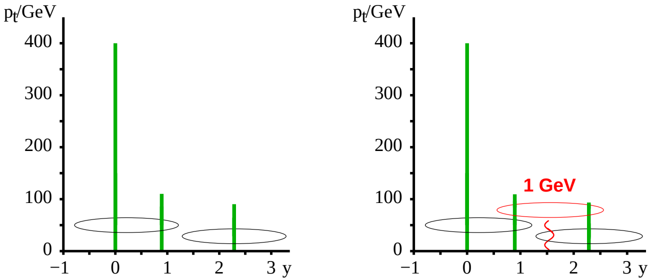
CDF Midpoint algorithm

D0 Run II algorithm

This solves problem for 2-hard-particle configs.

[But it persists for 3-hard]

Midpoint IR problem



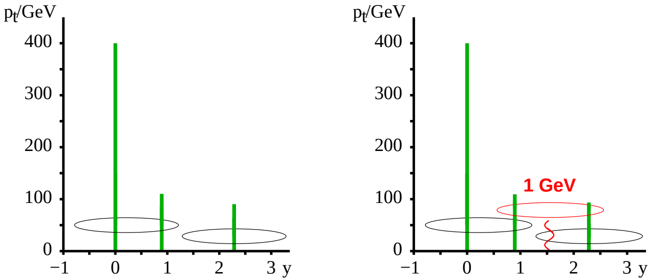
Stable cones with midpoint:	{1,2} & {3}	{1,2} & {2,3} & {3}
Jets with midpoint ($f = 0.5$)	{1,2} & {3}	{1,2,3}

Midpoint cone alg. misses some stable cones; extra soft particle → extra starting point → extra stable cone found

MIDPOINT IS INFRARED UNSAFE

Or collinear unsafe with seed threshold

Midpoint IR problem



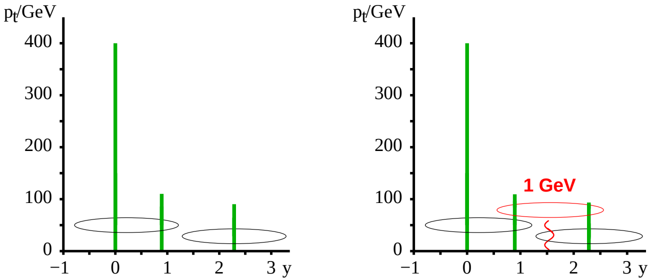
Stable cones with midpoint:	{1,2} & {3}	{1,2} & {2,3} & {3}
Jets with midpoint ($f = 0.5$)	{1,2} & {3}	{1,2,3}

Midpoint cone alg. misses some stable cones; extra soft particle → extra starting point → extra stable cone found

MIDPOINT IS INFRARED UNSAFE

Or collinear unsafe with seed threshold

Midpoint IR problem



Stable cones with midpoint:	{1,2} & {3}	{1,2} & {2,3} & {3}
Jets with midpoint ($f = 0.5$)	{1,2} & {3}	{1,2,3}

Midpoint cone alg. misses some stable cones; extra soft particle → extra starting point → extra stable cone found

MIDPOINT IS INFRARED UNSAFE

Or collinear unsafe with seed threshold

Does IRC safety really matter?

IRC safety & real-life

Real life does not have infinities, but pert. infinity leaves a real-life trace

$$\alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \infty \to \alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \ln p_t/\Lambda \to \alpha_s^2 + \underbrace{\alpha_s^3 + \alpha_s^3}_{\text{BOTH WASTED}}$$

Among consequences of IR unsafety:

	Last meaningful order			
	JetClu, ATLAS cone [IC-SM]	MidPoint [IC$_{mp}$-SM]	CMS it. cone [IC-PR]	Known at
Inclusive jets	LO	NLO	NLO	NLO (→ NNLO)
W/Z + 1 jet	LO	NLO	NLO	NLO
3 jets	**none**	LO	LO	NLO [nlojet++]
W/Z + 2 jets	**none**	LO	LO	NLO [MCFM]
m_{jet} in $2j + X$	**none**	**none**	**none**	LO

NB: \$30 – 50M investment in NLO

Multi-jet contexts much more sensitive: **ubiquitous at LHC**

And LHC will rely on QCD for background double-checks

extraction of cross sections, extraction of parameters

IRC safety & real-life

Real life does not have infinities, but pert. infinity leaves a real-life trace

$$\alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \infty \to \alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \ln p_t/\Lambda \to \alpha_s^2 + \underbrace{\alpha_s^3 + \alpha_s^3}_{\text{BOTH WASTED}}$$

Among consequences of IR unsafety:

	Last meaningful order			
	JetClu, ATLAS cone [IC-SM]	MidPoint [IC_{mp}-SM]	CMS it. cone [IC-PR]	Known at
Inclusive jets	LO	NLO	NLO	NLO ($\to$ NNLO)
W/Z + 1 jet	LO	NLO	NLO	NLO
3 jets	**none**	LO	LO	NLO [nlojet++]
W/Z + 2 jets	**none**	LO	LO	NLO [MCFM]
m_{jet} in $2j + X$	**none**	**none**	**none**	LO

NB: $30 – 50M investment in NLO

Multi-jet contexts much more sensitive: **ubiquitous at LHC**

And LHC will rely on QCD for background double-checks

extraction of cross sections, extraction of parameters

IRC safety & real-life

Real life does not have infinities, but pert. infinity leaves a real-life trace

$$\alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \infty \to \alpha_s^2 + \alpha_s^3 + \alpha_s^4 \times \ln p_t/\Lambda \to \alpha_s^2 + \underbrace{\alpha_s^3 + \alpha_s^3}_{\text{BOTH WASTED}}$$

Among consequences of IR unsafety:

	Last meaningful order			
	JetClu, ATLAS cone [IC-SM]	MidPoint [IC_{mp}-SM]	CMS it. cone [IC-PR]	Known at
Inclusive jets	LO	NLO	NLO	NLO ($\to$ NNLO)
W/Z + 1 jet	LO	NLO	NLO	NLO
3 jets	**none**	LO	LO	NLO [nlojet++]
W/Z + 2 jets	**none**	LO	LO	NLO [MCFM]
m_{jet} in $2j + X$	**none**	**none**	**none**	LO

NB: \$30 – 50M investment in NLO

Multi-jet contexts much more sensitive: **ubiquitous at LHC**

And LHC will rely on QCD for background double-checks

extraction of cross sections, extraction of parameters

IRC safety not just for theory

1. Detectors play tricks with soft particles

   calorimeter thresholds

   magnetic fields acting on charged particles

   calorimeter noise

2. Detectors split/merge collinear particles

   Two particles into single calo-tower

   One particles showers into two calo-towers

3. High lumi adds lots of extra soft seeds

IRC safety provides resilience to these effects

1 & 3 shift energy scale, but don't change overall jet-structure

If jet-algorithm is not IRC safe, fine-details of detector effects have potentially significant impact

IRC safety not just for theory

1. Detectors play tricks with soft particles

   calorimeter thresholds

   magnetic fields acting on charged particles

   calorimeter noise

2. Detectors split/merge collinear particles

   Two particles into single calo-tower

   One particles showers into two calo-towers

3. High lumi adds lots of extra soft seeds

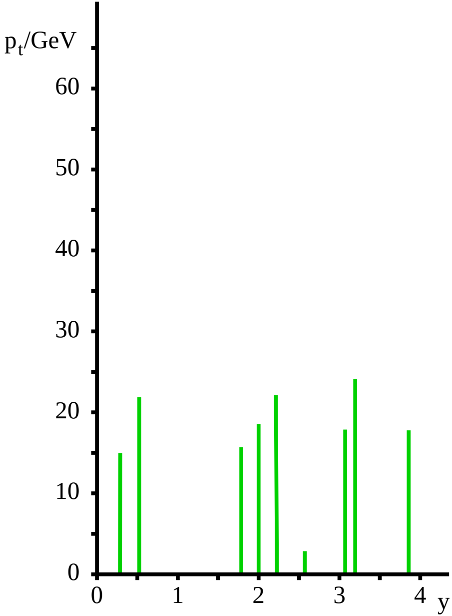
IRC safety provides resilience to these effects

1 & 3 shift energy scale, but don't change overall jet-structure

If jet-algorithm is not IRC safe, fine-details of detector effects have potentially significant impact

Can we cure this IR safety problem?

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

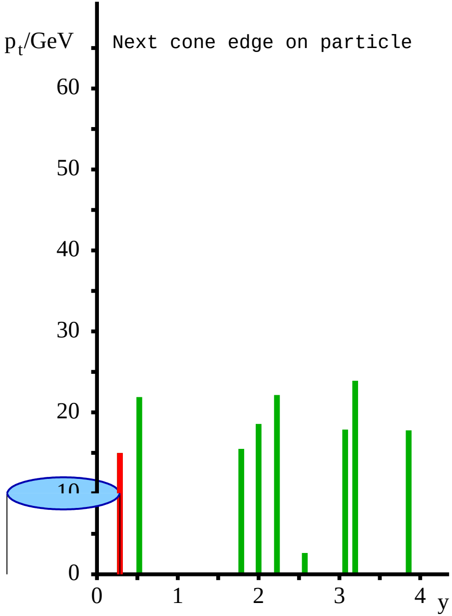
Procedure in 1 dimension (y):

- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**
GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

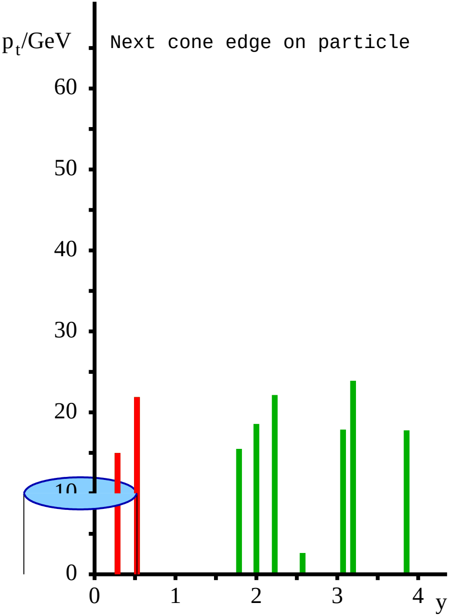
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

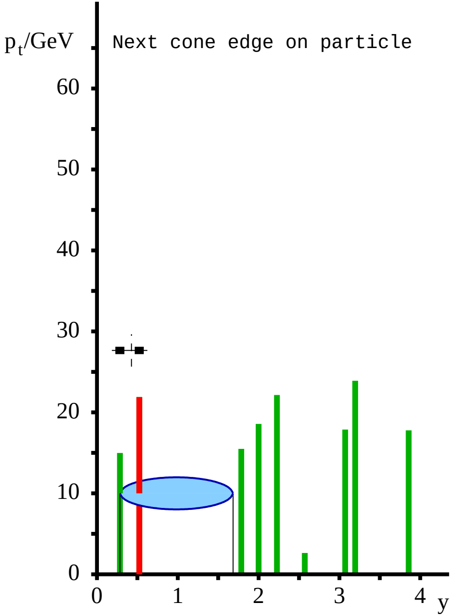
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

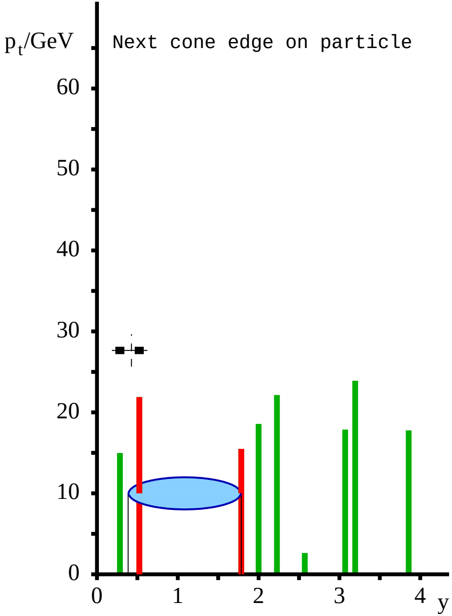
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure — **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

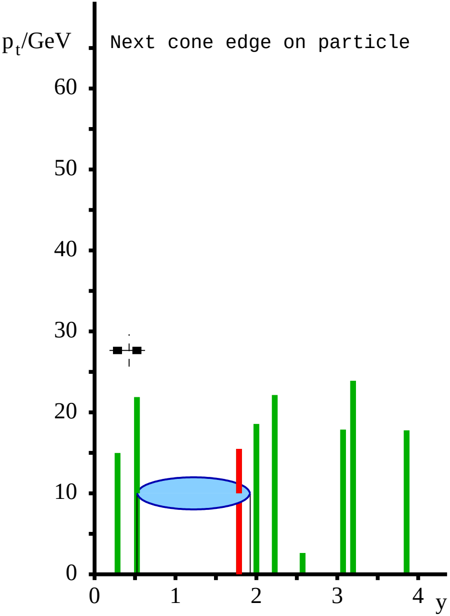
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

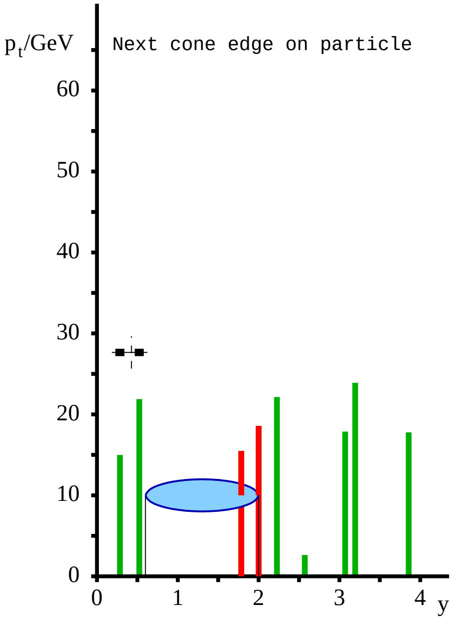
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) *can design analogous procedure* **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

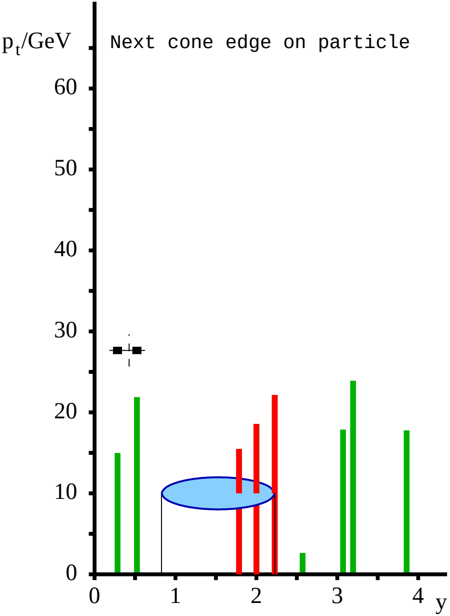
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

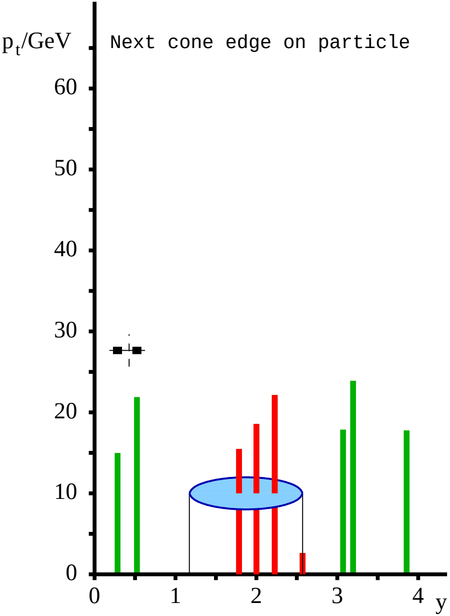
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) *can design analogous procedure* **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

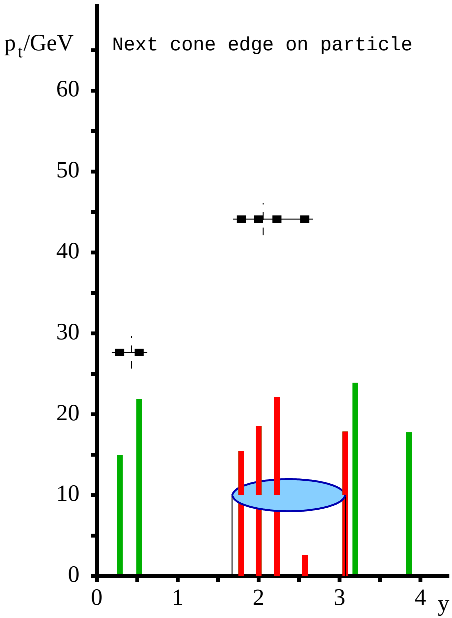
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

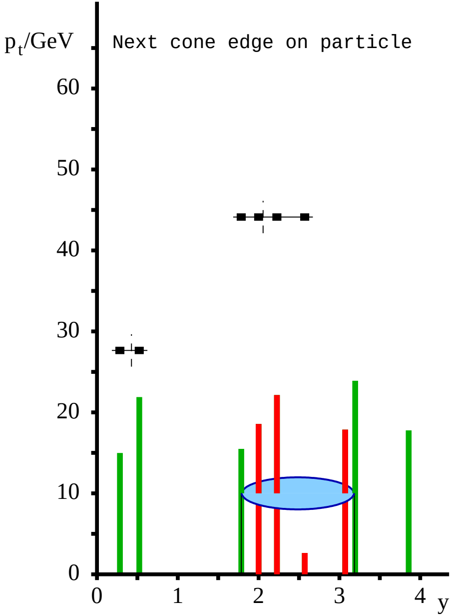
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

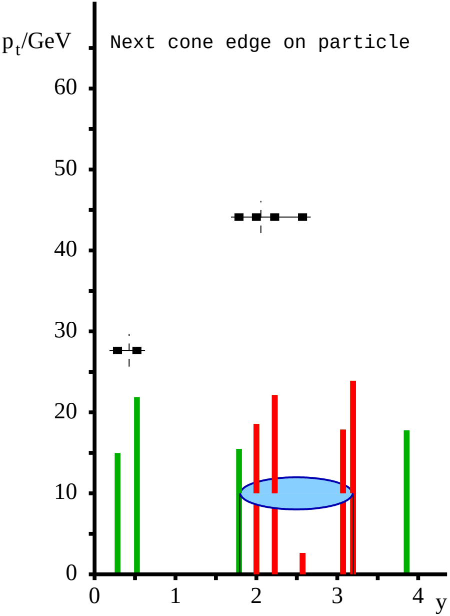
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure — **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

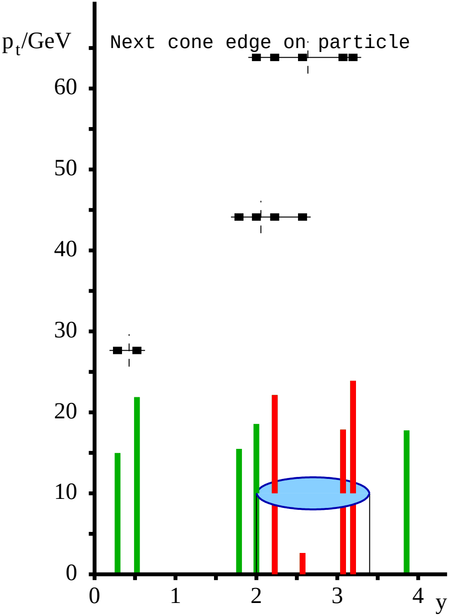
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure — **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

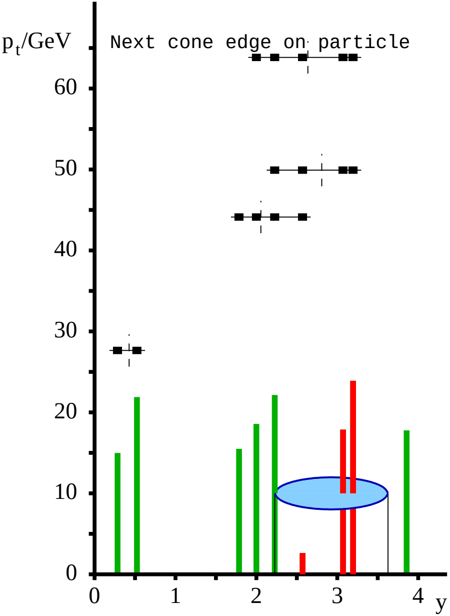
Procedure in 1 dimension (y):

- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure — **SISCone**
GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

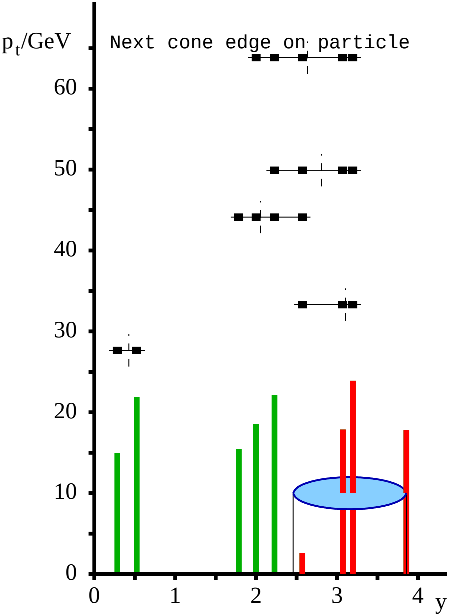
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure — **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

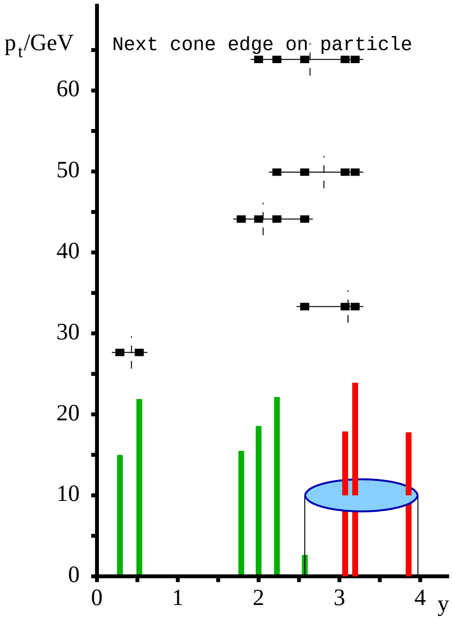
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) *can design analogous procedure* **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

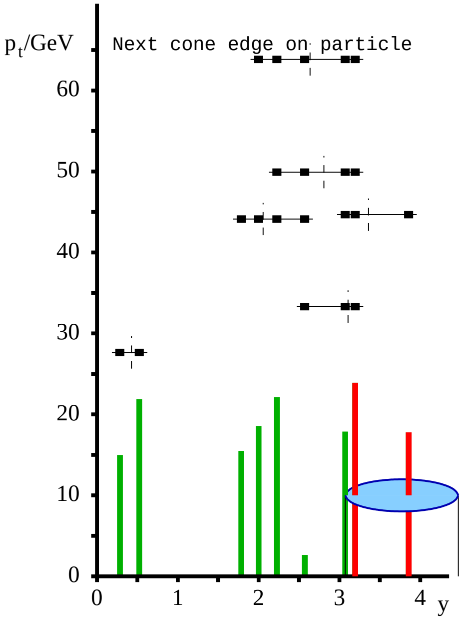
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

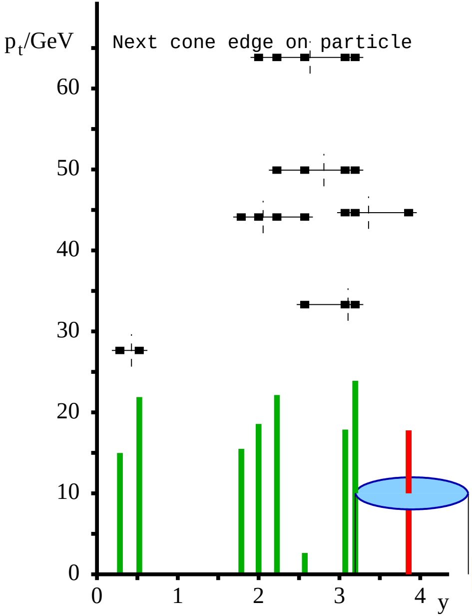
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) *can design analogous procedure* **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

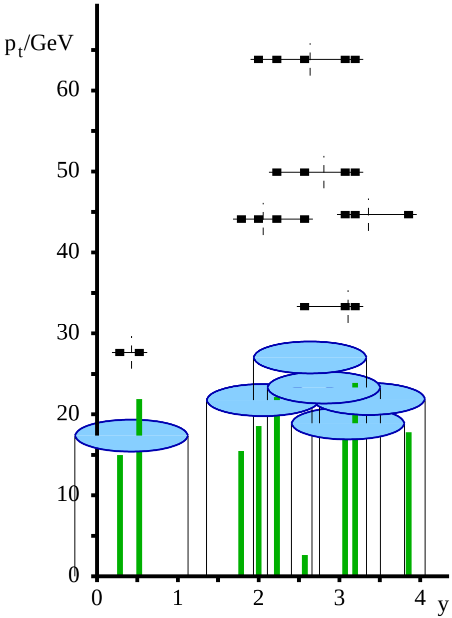
- find all distinct enclosures of radius R by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

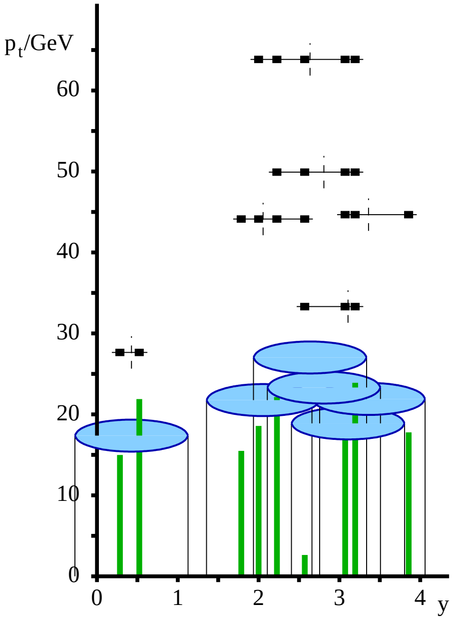
- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y,ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Seedless [Infrared Safe] cones (SC-SM)



Aim to identify *all* stable cones, independently of any seeds

Procedure in 1 dimension (y):

- find all distinct enclosures of radius R — by repeatedly sliding a cone sideways until edge touches a particle
- check each for stability
- then run usual split–merge

In 2 dimensions (y, ϕ) can design analogous procedure **SISCone**

GPS & Soyez '07

This gives an IRC safe cone alg.

Is it truly IR safe?

- Generate event with $2 < N < 10$ hard particles, find jets
- Add $1 < N_{soft} < 5$ soft particles, find jets again [repeatedly]
- If the jets are different, algorithm is IR unsafe.

Unsafety level	failure rate
2 hard + 1 soft	$\sim 50\%$
3 hard + 1 soft	$\sim 15\%$
SISCone	IR safe !

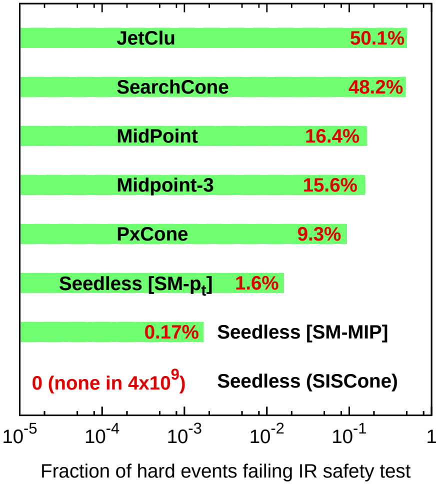
Be careful with split–merge too

Is it truly IR safe?

- Generate event with $2 < N < 10$ hard particles, find jets
- Add $1 < N_{soft} < 5$ soft particles, find jets again [repeatedly]
- If the jets are different, algorithm is IR unsafe.

Unsafety level	failure rate
2 hard + 1 soft	$\sim$ 50%
3 hard + 1 soft	$\sim$ 15%
SISCone	IR safe !

Be careful with split–merge too



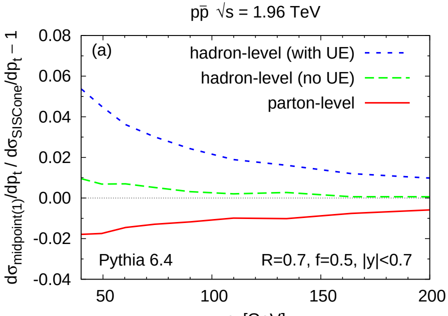
How much does IR safety *really* matter?

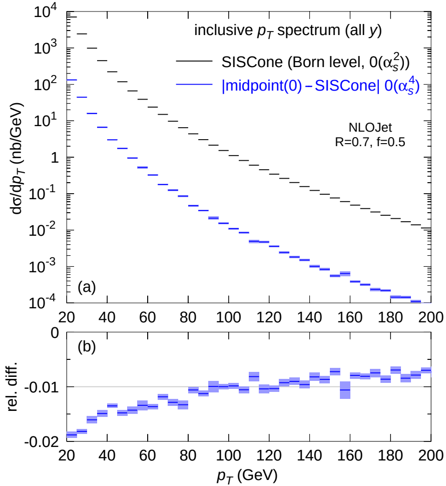
Compare midpoint and SISCone

Result depends on observable:

- inclusive jet spectrum is the least sensitive (affected at NNLO)
- larger differences (5 – 10%) at hadron level

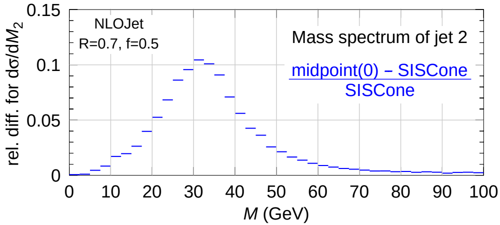
seedless reduces UE effect





IR safety & multi-jet observables

Look at jet masses in multijet events. NB: Jet masses reconstruct boosted $W/Z/H/top$ in BSM searches



Select 3-jet events

$$p_{t1,2,3} > \{120, 60, 20\} \text{ GeV},$$

Calculate LO jet-mass spectrum for jet 2, compare midpoint with SISCone.

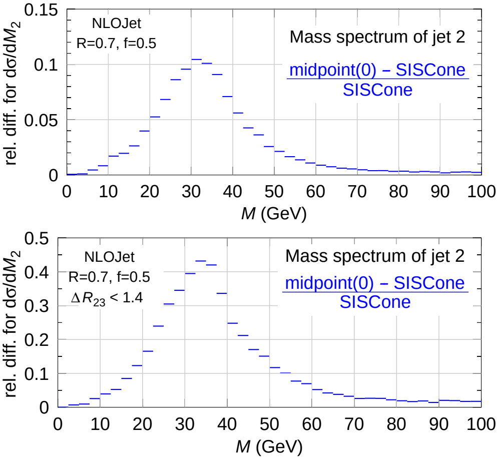
- 10% differences by default
- **40% differences** with extra cut $\Delta R_{2,3} < 1.4$

  e.g. for jets from common decay chain

In complex events, IR safety matters

IR safety & multi-jet observables

Look at jet masses in multijet events. NB: Jet masses reconstruct boosted $W/Z/H/top$ in BSM searches



Select 3-jet events

$p_{t1,2,3} > \{120, 60, 20\}$ GeV,

Calculate LO jet-mass spectrum for jet 2, compare midpoint with SISCone.

- 10% differences by default
- **40% differences** with extra cut $\Delta R_{2,3} < 1.4$

e.g. for jets from common decay chain

In complex events, IR safety matters

Bottom line on IR safety

- IR safety often matters less in *inclusive* quantities
- It matters more in multi-jet cases

- ATLAS cone, JetClu (IC-SM) are *very bad*
- CMS cone (IC-PR), Midpoint (IC_{mp}-SM) *moderately bad*

- An IRC safe cone algorithm exists (SISCone)
- **Avoid trouble later: use IR-safe algs from the start**

cf. CDF W+jets

What jet definition should I use?

[jet def. $\equiv$ jet alg., R, (f)]

A full set of IRC-safe jet algorithms

Generalise inclusive-type sequential recombination with

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

	Alg. name	Comment	time
$p = 1$	k_t CDOSTW '91-93; ES '93	Hierarchical in rel. k_t	$N \ln N$ exp.
$p = 0$	Cambridge/Aachen Dok, Leder, Moretti, Webber '97 Wengler, Wobisch '98	Hierarchical in angle Scan multiple R at once $\leftrightarrow$ QCD angular ordering	$N \ln N$
$p = -1$	anti-k_t Cacciari, GPS, Soyez '08 $\sim$ reverse-k_t Delsart	Hierarchy meaningless, jets like CMS cone (IC-PR)	$N^{3/2}$
SC-SM	SISCone GPS Soyez '07 + Tevatron run II '00	Replaces JetClu, ATLAS MidPoint (xC-SM) cones	$N^2 \ln N$ exp.

Compromise between having a limited set of algs.
and a good range of complementary properties

COMMERCIAL BREAK

Use FastJet — it's free!

One place to stop for all your jet-finding needs:

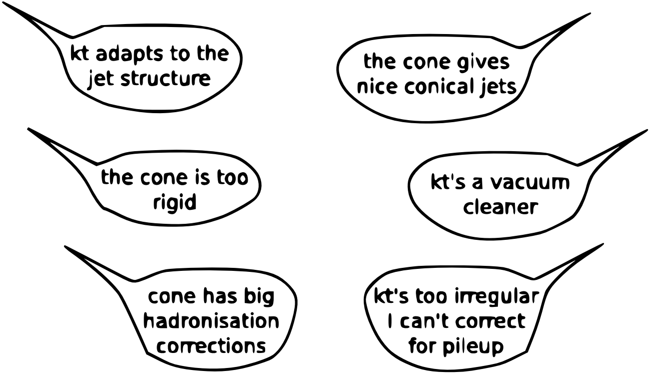
FASTJET

http://www.lpthe.jussieu.fr/~salam/fastjet
Cacciari, GPS & Soyez '05–07

- Fast, native, computational-geometry methods for k_t, Cam/Aachen
  Cacciari & GPS '05-06
- Plugins for SISCone (plus some other, deprecated cones)
- Many other features too, e.g. jet areas

Jet Folklore

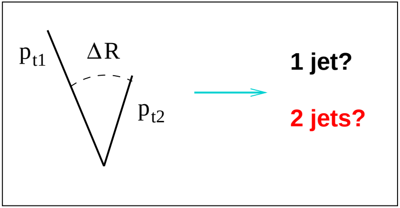
Jet discussions: often polarised, driven by unquantified statements



- Rigorous approach is to quantify similarities & differences
- Bottom line: grains of truth in the qualitative statements

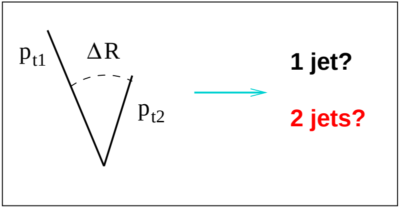
  So want good cone algorithms too [NB: recall, two variants xC-SM & xC-PR]

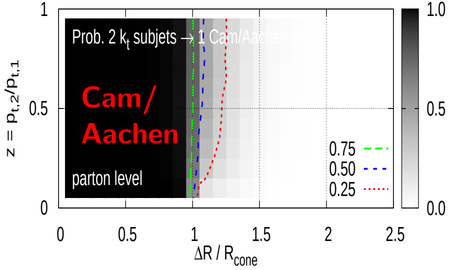
the *reach* of jet algorithms



SISCone (xC-SM) reaches further for hard radiation than other algs

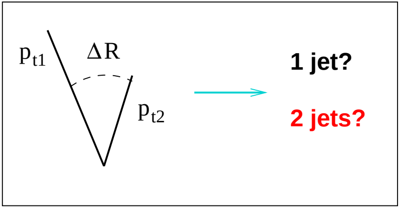
the *reach* of jet algorithms

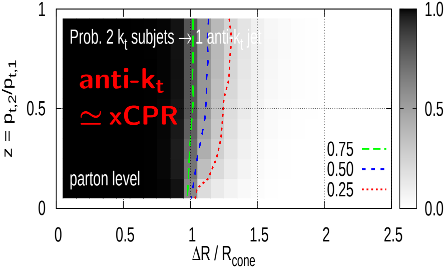


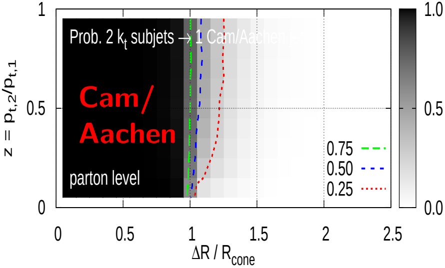


SISCone (xC-SM) reaches further for hard radiation than other algs

the *reach* of jet algorithms

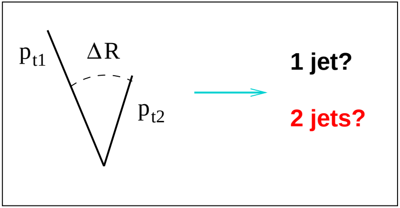


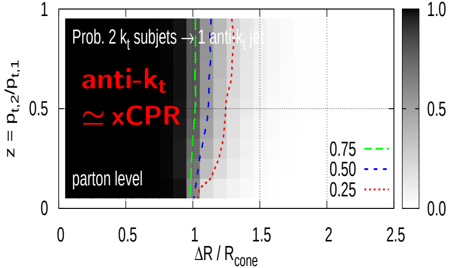


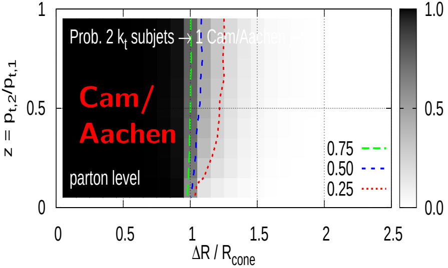


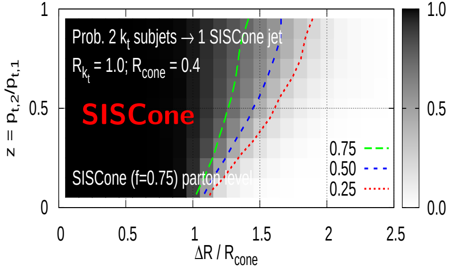
SISCone (xC-SM) reaches further for hard radiation than other algs

the *reach* of jet algorithms



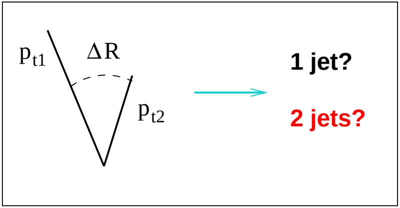


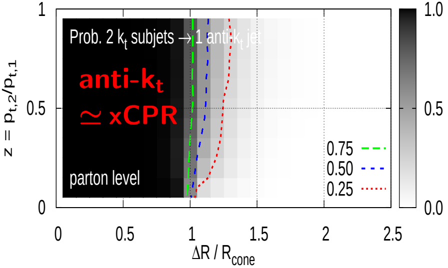


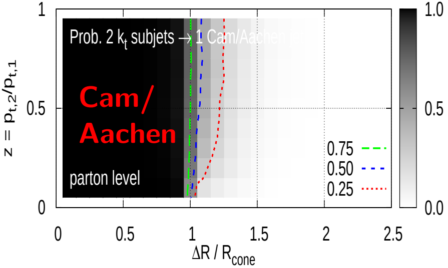


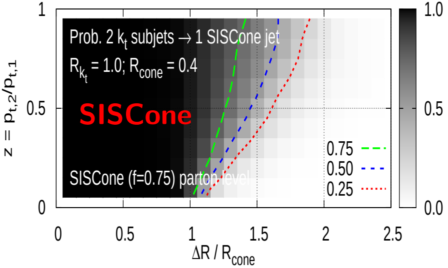
SISCone (xC-SM) reaches further for hard radiation than other algs

the *reach* of jet algorithms



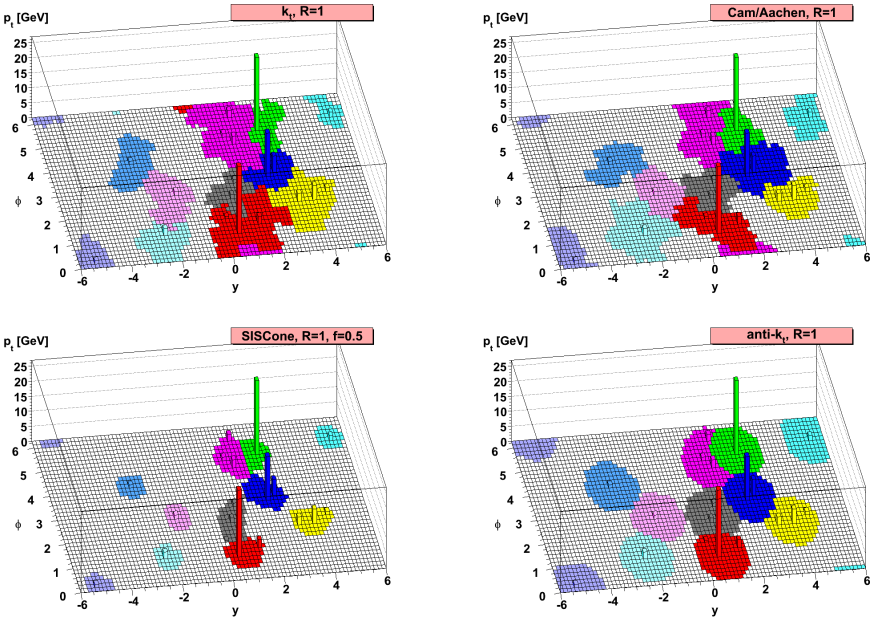






SISCone (xC-SM) reaches further for hard radiation than other algs

Jet contours – visualised



To first approx:
various algs. moderately different;
but **R** can matter a lot more

4-way tension in many measurements:

Prefer small R	prefer large R
resolve many jets (e.g. $t\bar{t}$)	minimize QCD radiation loss
limit UE & pileup	limit hadronisation

Jets v. R

Parton $p_t \to$ jet p_t

Ill-defined: MC "parton"

PT radiation:

$$q: \quad \Delta p_t \simeq \frac{\alpha_s C_F}{\pi} p_t \ln R$$

$$g: \quad \Delta p_t \simeq \frac{\alpha_s C_A}{\pi} p_t \ln R$$

Hadronisation:

$$q: \quad \Delta p_t \simeq \frac{C_F}{R} \cdot 0.4 \text{ GeV}$$

$$g: \quad \Delta p_t \simeq \frac{C_A}{R} \cdot 0.4 \text{ GeV}$$

Underlying event:

$$q, g: \quad \Delta p_t \simeq \frac{R^2}{2} \cdot 2.5\text{–}15 \text{ GeV}$$

crude analytical estimates
cf. Dasgupta, Magnea & GPS '07

Jets v. R

Parton $p_t \longrightarrow$ jet p_t

Ill-defined: MC "parton"

PT radiation:

$$q: \quad \Delta p_t \simeq \frac{\alpha_s C_F}{\pi} p_t \ln R$$

$$g: \quad \Delta p_t \simeq \frac{\alpha_s C_A}{\pi} p_t \ln R$$

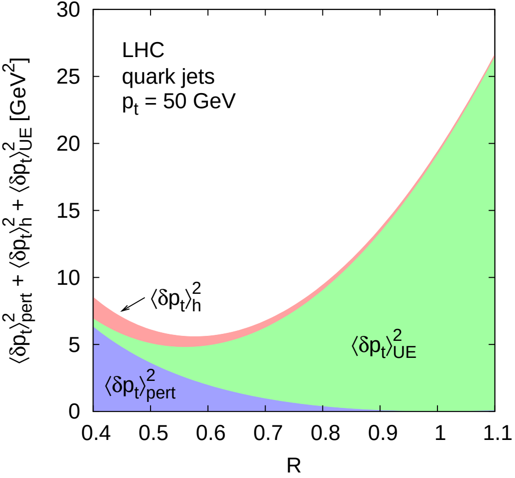
Hadronisation:

$$q: \quad \Delta p_t \simeq \frac{C_F}{R} \cdot 0.4 \text{ GeV}$$

$$g: \quad \Delta p_t \simeq \frac{C_A}{R} \cdot 0.4 \text{ GeV}$$

Underlying event:

$$q, g: \quad \Delta p_t \simeq \frac{R^2}{2} \cdot 2.5\text{—}15 \text{ GeV}$$



crude analytical estimates

cf. Dasgupta, Magnea & GPS '07

Jets v. R

Parton $p_t \longrightarrow$ jet p_t

Ill-defined: MC "parton"

PT radiation:

$$q: \quad \Delta p_t \simeq \frac{\alpha_s C_F}{\pi} p_t \ln R$$

$$g: \quad \Delta p_t \simeq \frac{\alpha_s C_A}{\pi} p_t \ln R$$

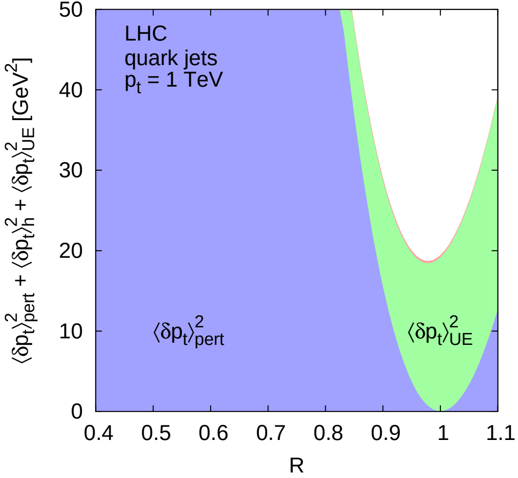
Hadronisation:

$$q: \quad \Delta p_t \simeq \frac{C_F}{R} \cdot 0.4 \text{ GeV}$$

$$g: \quad \Delta p_t \simeq \frac{C_A}{R} \cdot 0.4 \text{ GeV}$$

Underlying event:

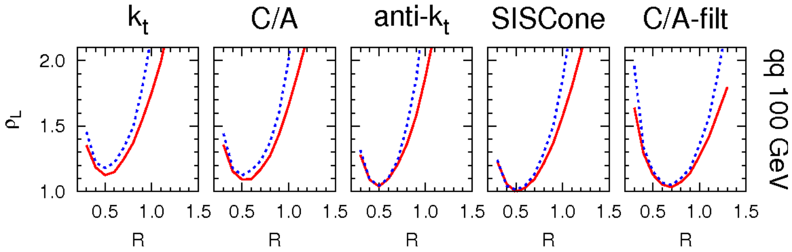
$$q, g: \quad \Delta p_t \simeq \frac{R^2}{2} \cdot 2.5{-}15 \text{ GeV}$$



crude analytical estimates

cf. Dasgupta, Magnea & GPS '07

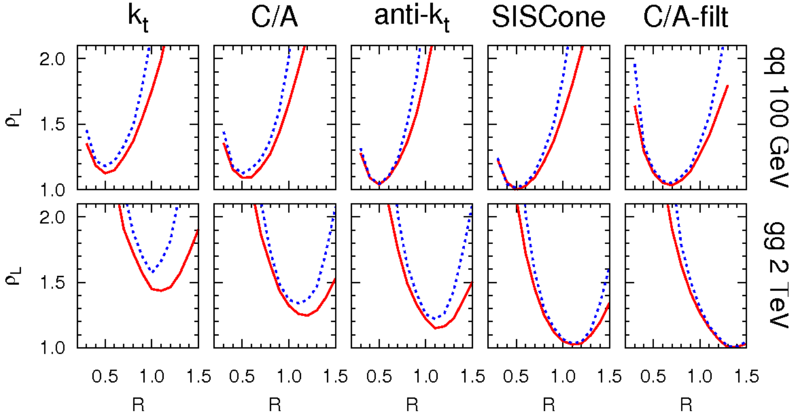
Relative peak quality (lumi ratios ρ_L), LHC



PRELIMINARY

Cacciari, Rojo, GPS & Soyez '08

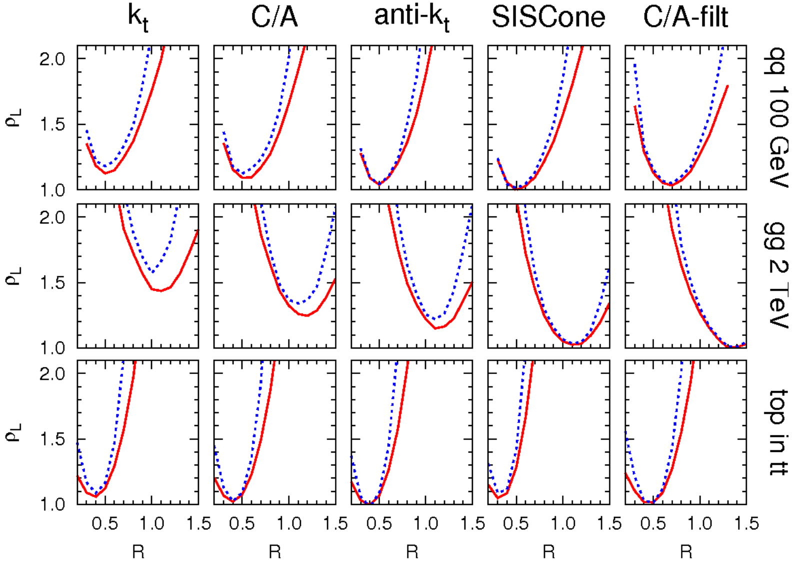
Relative peak quality (lumi ratios ρ_L), LHC



PRELIMINARY

Cacciari, Rojo, GPS & Soyez '08

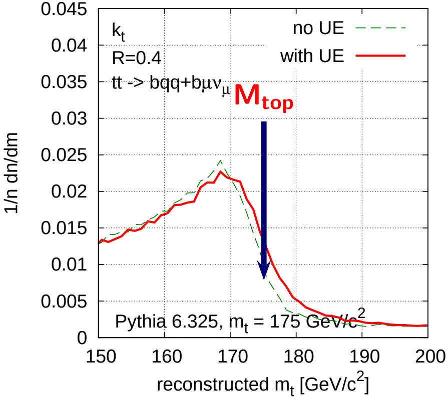
Relative peak quality (lumi ratios ρ_L), LHC



PRELIMINARY

Cacciari, Rojo, GPS & Soyez '08

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

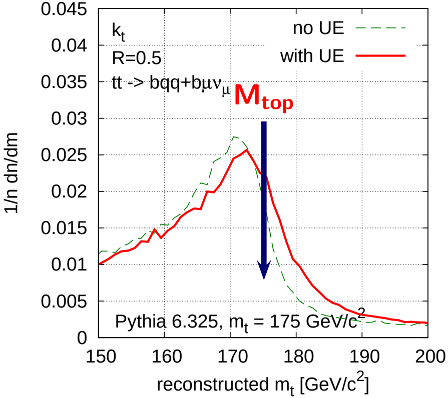
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

Is the final top mass (after W, jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

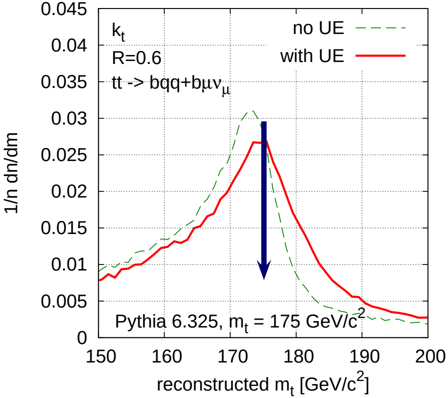
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

Is the final top mass (after W, jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

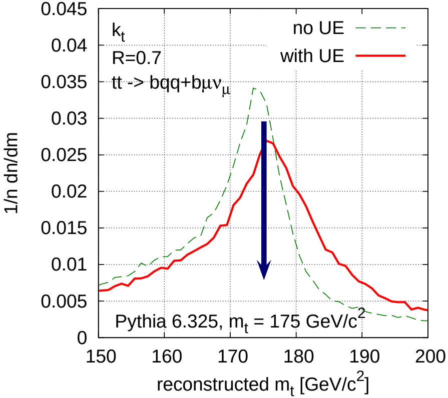
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2 - 4$ GeV.

Is the final top mass (after W jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

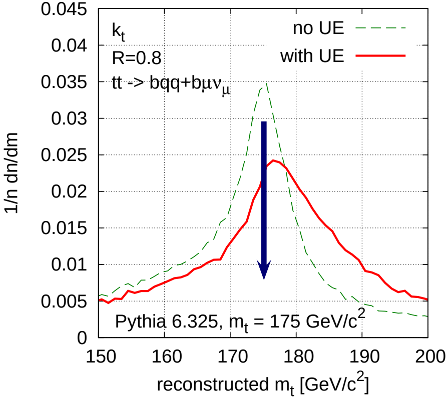
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2 - 4$ GeV.

Is the final top mass (after W jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

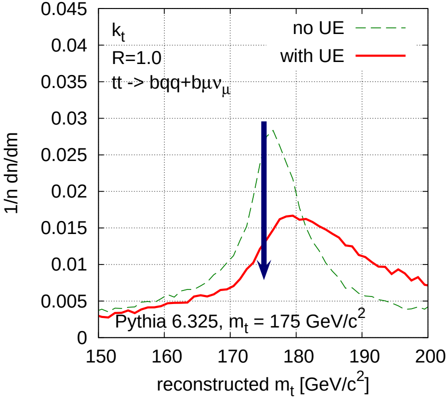
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

Is the final top mass (after W jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Robustness: M_{top} varies with R?



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

Is the final top mass (after W jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

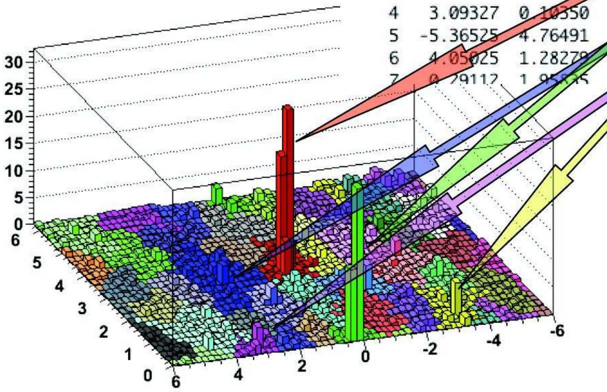
Jets without hard partons:

Most jet algorithms give you $\sim 50 - 100$ “jets,” mostly not hard.

provide window on UE and min-bias

Making use of *all* jets

```
iev 0 (irepeat 24): number of particles = 1428
strategy used =  NlnN
number of particles = 9051
Total area: 76.0265
Expected area: 76.0265
 ijet    eta        phi          Pt          area  +-    err
    0   0.15050   3.24498      69.970       2.625 +-   0.020
    1   0.18579   0.13150      59.133       1.896 +-   0.020
    2   2.33840   3.23960      31.976       4.749 +-   0.028
    3  -3.41796   0.52394      26.595       3.084 +-   0.021
    4   3.09327   0.10350      20.072       2.688 +-   0.023
    5  -5.36525   4.76491      19.593       2.780 +-   0.012
    6   4.05025   1.28279      15.561       3.592 +-   0.028
    7   0.29112   1.95835      14.566       2.114 +-   0.018
```



Approximate linear relation between Pt and area for minimum bias jets.

Can be used on an event-by-event basis to correct the hard jets

Pushing jets to their limit:

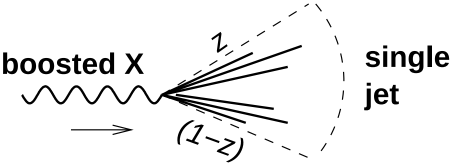
when a W, Z, H or a top $\longrightarrow$ a single jet

Not unusual at LHC: $m_W, m_t \ll 14$ TeV

EW bosons at @ high p_t

Illustrate LHC challenges with a recently widely discussed class of problems:

Can you identify hadronically decaying EW bosons when they're produced at high p_t?



$$R \gtrsim \frac{m}{p_t}\frac{1}{\sqrt{z(1-z)}}$$

Significant discussion over years: **heavy new things decay to EW states**

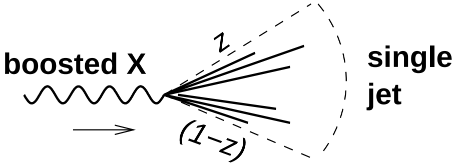
- Seymour '94 [Higgs $\to WW \to \nu\ell$jets]
- Butterworth, Cox & Forshaw '02 [$WW \to WW \to \nu\ell$jets]
- Agashe et al. '06 [KK excitation of gluon $\to t\bar{t}$]
- Butterworth, Ellis & Raklev '07 [SUSY decay chains $\to W, H$]
- Skiba & Tucker-Smith '07 [vector quarks]
- Lilli, Randall & Wang '07 [KK excitation of gluon $\to t\bar{t}$]

ETC.

EW bosons at @ high p_t

Illustrate LHC challenges with a recently widely discussed class of problems:

Can you identify hadronically decaying EW bosons when they're produced at high $\mathbf{p_t}$?



$$R \gtrsim \frac{m}{p_t} \frac{1}{\sqrt{z(1-z)}}$$

Significant discussion over years: **heavy new things decay to EW states**

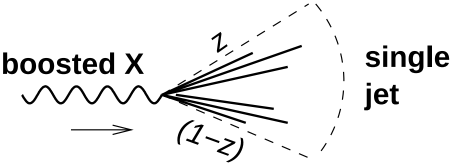
- Seymour '94 [Higgs $\to WW \to \nu\ell$jets]
- Butterworth, Cox & Forshaw '02 [$WW \to WW \to \nu\ell$jets]
- Agashe et al. '06 [KK excitation of gluon $\to t\bar{t}$]
- Butterworth, Ellis & Raklev '07 [SUSY decay chains $\to W, H$]
- Skiba & Tucker-Smith '07 [vector quarks]
- Lilli, Randall & Wang '07 [KK excitation of gluon $\to t\bar{t}$]

ETC.

EW bosons at @ high p_t

Illustrate LHC challenges with a recently widely discussed class of problems:

Can you identify hadronically decaying EW bosons when they're produced at high p_t?



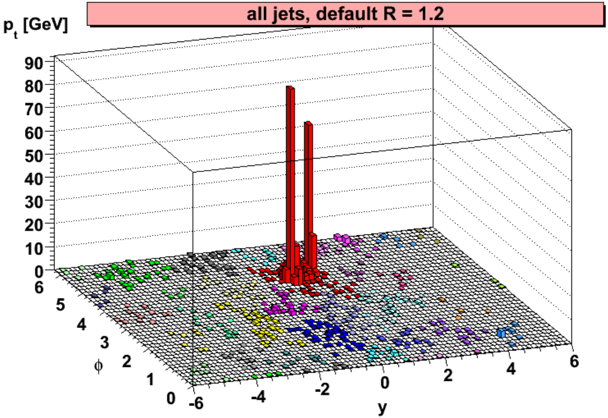
$$R \gtrsim \frac{m}{p_t} \frac{1}{\sqrt{z(1-z)}}$$

Significant discussion over years: **heavy new things decay to EW states**

- Seymour '94 [Higgs $\to WW \to \nu\ell$jets]
- Butterworth, Cox & Forshaw '02 [$WW \to WW \to \nu\ell$jets]
- Agashe et al. '06 [KK excitation of gluon $\to t\bar{t}$]
- Butterworth, Ellis & Raklev '07 [SUSY decay chains $\to W, H$]
- Skiba & Tucker-Smith '07 [vector quarks]
- Lilli, Randall & Wang '07 [KK excitation of gluon $\to t\bar{t}$]

ETC.

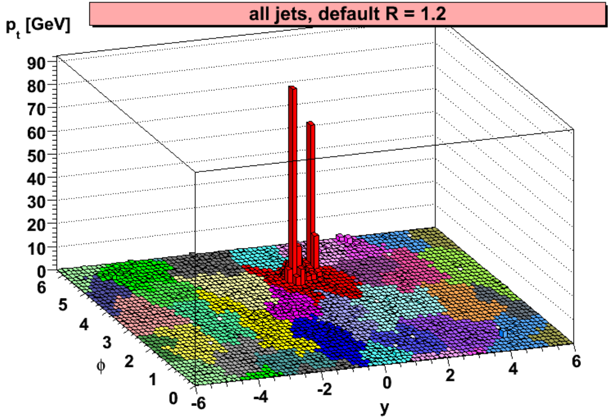
$pp \rightarrow ZH \rightarrow \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV



[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]
Butterworth, Davison, Rubin & GPS '08

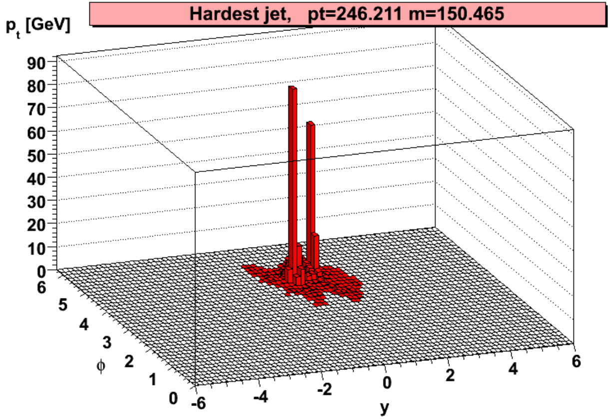
Possible new (light) Higgs discovery channel

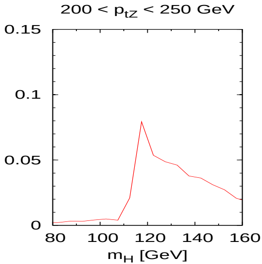
$pp \to ZH \to \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV

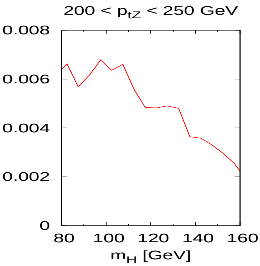


[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]
Butterworth, Davison, Rubin & GPS '08

$pp \longrightarrow ZH \longrightarrow \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV



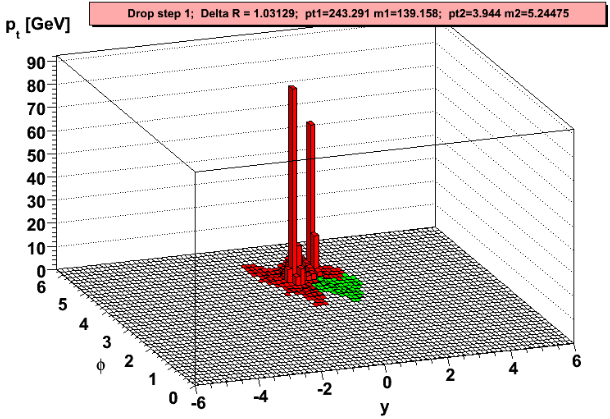




[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

Butterworth, Davison, Rubin & GPS '08

$pp \to ZH \to \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV

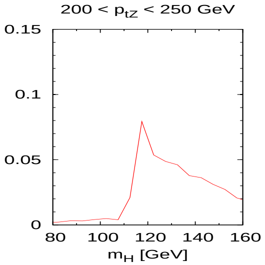


[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

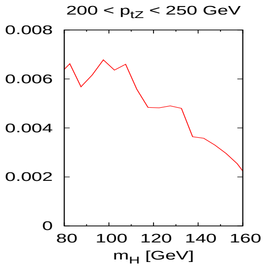
Butterworth, Davison, Rubin & GPS '08

Possible new (light) Higgs discovery channel

SIGNAL

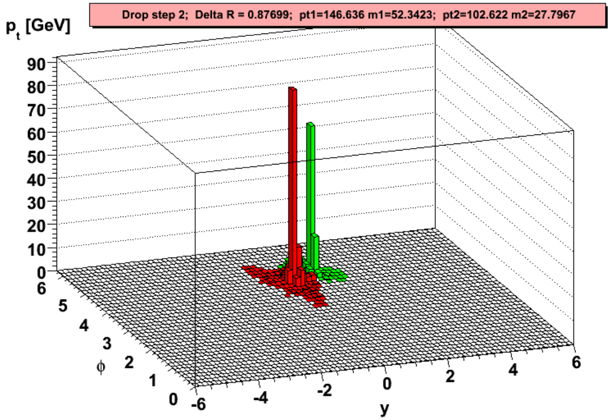


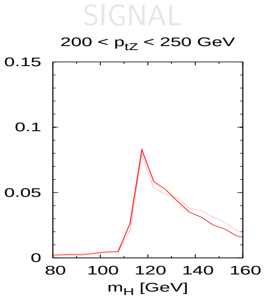
Zbb BACKGROUND

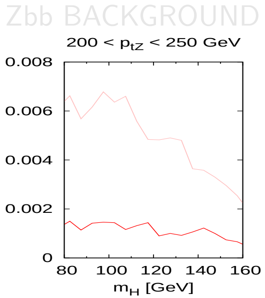


arbitrary norm.

$pp \longrightarrow ZH \longrightarrow \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV







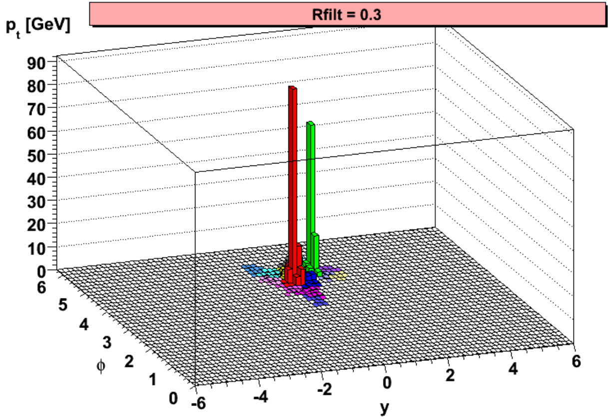
[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

Butterworth, Davison, Rubin & GPS '08

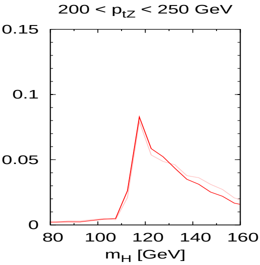
Possible new (light) Higgs discovery channel

arbitrary norm.

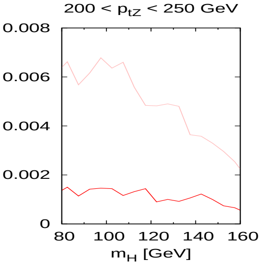
$pp \to ZH \to \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV



SIGNAL



Zbb BACKGROUND



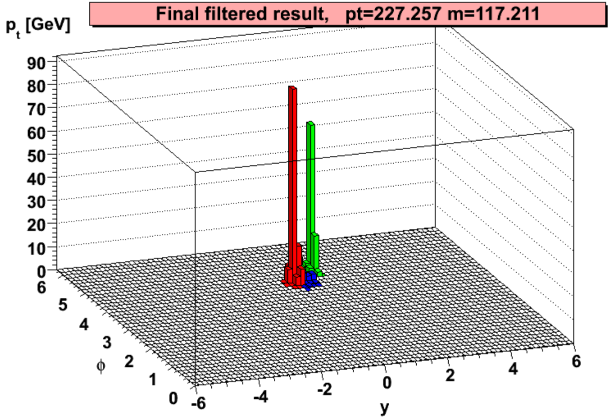
[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

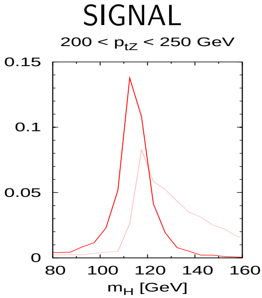
Butterworth, Davison, Rubin & GPS '08

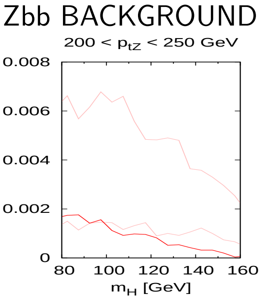
Possible new (light) Higgs discovery channel

arbitrary norm.

$pp \to ZH \to \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV







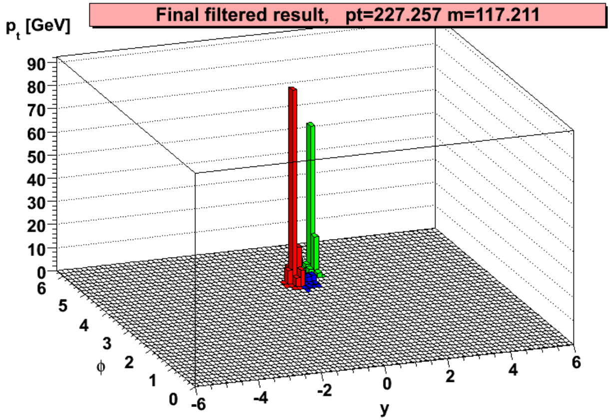
[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

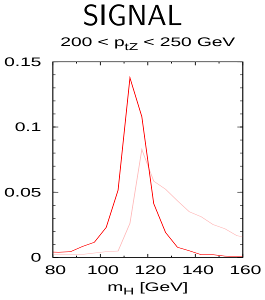
Butterworth, Davison, Rubin & GPS ’08

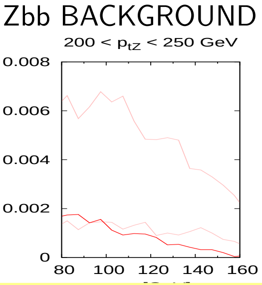
Possible new (light) Higgs discovery channel

arbitrary norm.

$pp \longrightarrow ZH \longrightarrow \nu\bar{\nu}b\bar{b}$, @14 TeV, $m_H = 115$ GeV







[Herwig 6.5 + Jimmy 4.31 + FastJet Cam/Aa R=1.2]

Butterworth, Davison, Rubin & GPS '08

Possible new (light) Higgs discovery channel

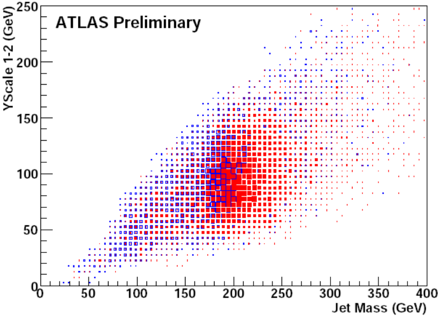
Much to be learnt still about extracting boosted W/H/Z/top?

High-p_t top $\longrightarrow b q \bar{q}'$

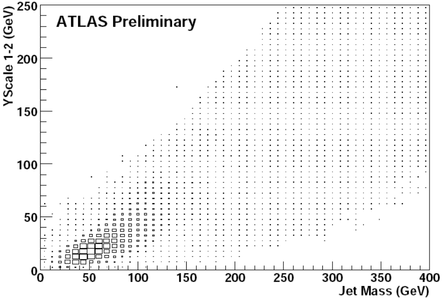
Brooijmans '08 ATL-PHYS-CONF-2008-008, based on k_t algorithm
+ Thaler & Wang '08; Almeida et al. '08 (k_t, jet-shapes)
+ Kaplan et al '08 (C/A decomposition)

Use subjet relative transverse-momentum scale ("'y-scale") & correlation with jet mass to pick out top quarks from background

top quarks $p_t \sim 1$ TeV



normal jets

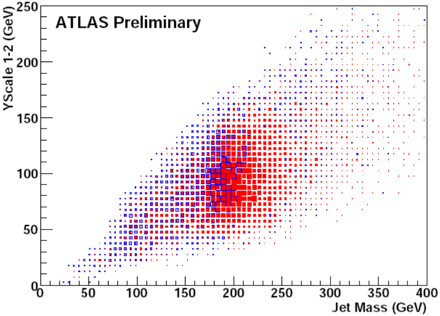


High-p_t top $\longrightarrow b q \bar{q}'$

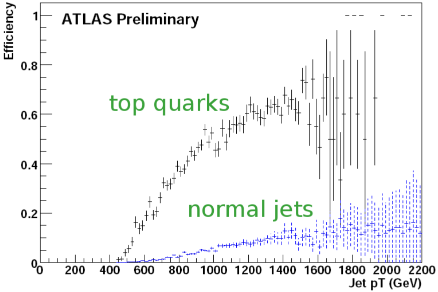
Brooijmans '08 ATL-PHYS-CONF-2008-008, based on k_t algorithm
+ Thaler & Wang '08; Almeida et al. '08 (k_t, jet-shapes)
+ Kaplan et al '08 (C/A decomposition)

Use subjet relative transverse-momentum scale ("'y-scale") & correlation with jet mass to pick out top quarks from background

top quarks $p_t \sim 1$ TeV



efficiencies



Conclusions

Conclusions

- A jet is not a parton: it's (sort of) what you choose it to be.
- It's easier to think in terms of partons (LO, NLO pQCD) with IR/Collinear safe jet algorithms. And gives sense to pQCD predictions
- ∃ many cones algs. Not equivalent. Many are IR/Coll unsafe.
  xC-SM → SISCone; xC-PR → anti-k_t
- "The best" jet definition *does not exist*
- To get the most out of jet-algs.,
  - Understand the interplay of physical scales high $p_t \to$ larger R
  - Try out different combinations of algorithm & R
  - Check Variations of alg. & R don't change extracted physical quantities
- Special cases (e.g. boosted W/t/. . .) benefit from special techniques
  e.g. seq. recomb. "jet-decomposition" is a powerful tool