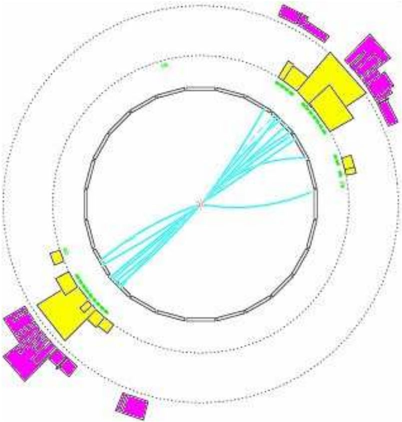
La théorie des jets

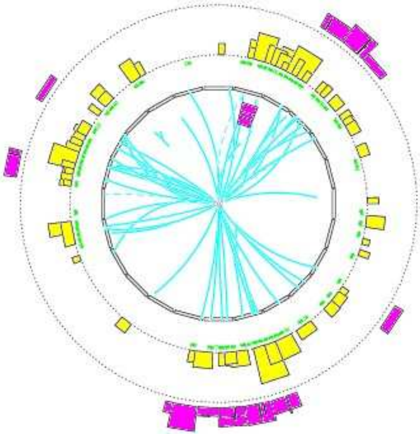
Gavin Salam

LPTHE, Universities of Paris VI and VII and CNRS

École de Gif 2007 — QCD et le LHC
24 au 28 Septembre 2007 LPNHE, Paris

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

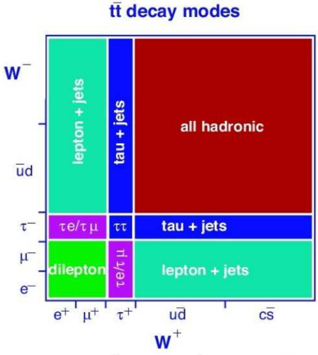
Non-perturbatively

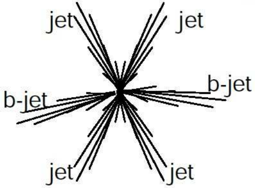
- precise process long way from being understood, even by lattice
- good models contain many parameters — complex process

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

See lecture by Skands

Jets at LHC





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

picture: Juste LP05

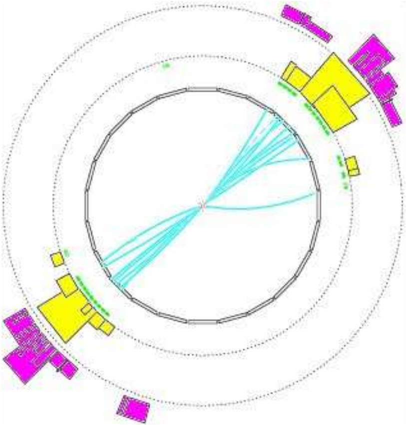
Heavy objects: multi-jet final-states

- 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, $\theta_{ij} > 30\,\text{deg}$, $|y_{ij}| < 3$
Draggiotis, Kleiss & Papadopoulos '02

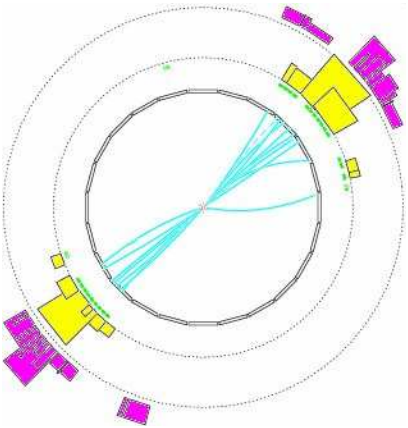
Seeing v. defining jets



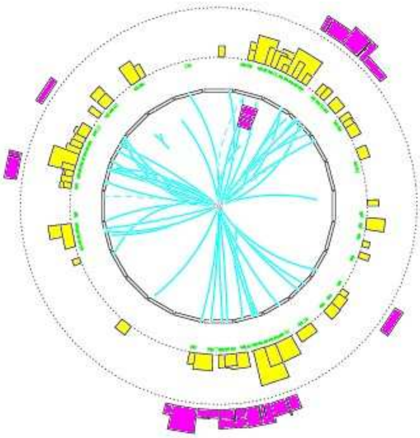
Jets are what we see.
Clearly(?) 2 jets here

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

Seeing v. defining jets

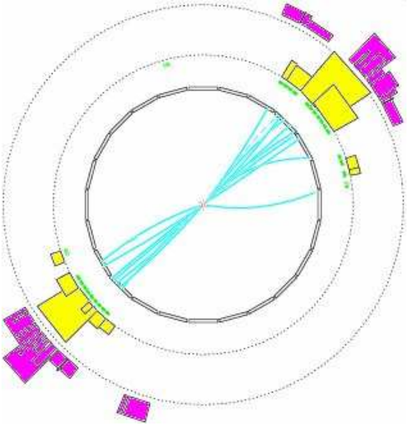


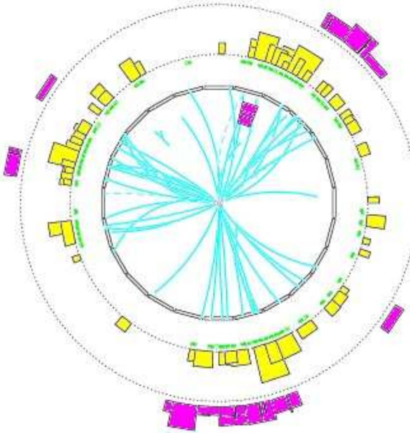
Jets are what we see.
Clearly(?) 2 jets here



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

Seeing v. defining jets





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

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

Jet definitions

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

This lecture

Aim: to provide an introduction to the "basics" you should be aware of if you carry out or review a hadron-collider analysis that uses jets.

- General considerations
- Common jet definitions
- Jets at work

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

Jets: like photography, vary focus

Jets: like photography, vary focus

Jets: like photography, vary focus

Jets: like photography, vary focus

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 → 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**

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

k_t/Durham algorithm

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

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

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

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

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

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

NB: relative k_t between particles

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

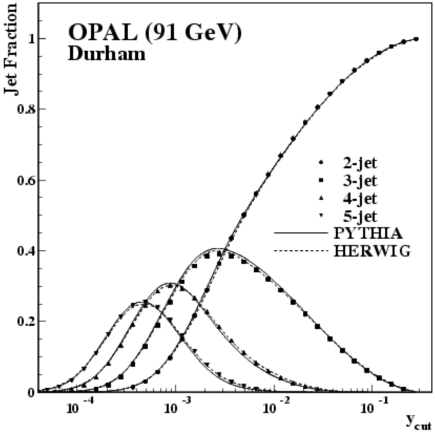
Catani, Dokshitzer, Olsson, Turnock & Webber '91

k_t/Durham algorithm features

- Gives hierarchy to event and jets

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

- Resolution parameter related to minimal transverse momentum between jets



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

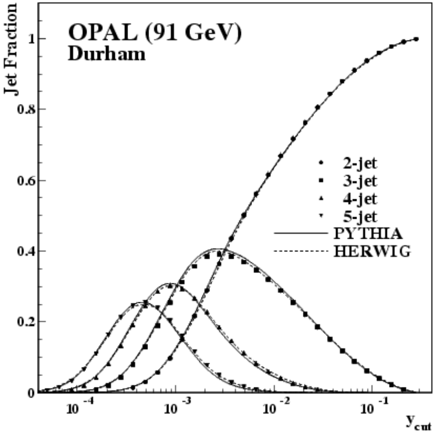
- Collinear safe: collinear particles recombined early on
- Infrared safe: soft particles have no impact on rest of clustering seq.

k_t/Durham algorithm features

- Gives hierarchy to event and jets

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

- Resolution parameter related to minimal transverse momentum between jets



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

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

k_t alg. at hadron colliders

1st attempt

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

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

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

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

squared transv. mom. wrt beam

k_t alg. at hadron colliders

2nd attempt: make it longitudinally boost-invariant

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

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

NB: not η_i, E_{ti}

- Beam distance becomes

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

squared transv. mom. wrt beam

Catani, Dokshitzer, Seymour & Webber '93

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

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

See e.g. Seymour & Tevlin '06

k_t alg. at hadron colliders

3nd attempt: **inclusive k_t algorithm**

- Introduce angular radius R (NB: dimensionless!)

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

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

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

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

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

k_t is a form of Hierarchical Clustering

Fast Hierarchical Clustering and Other Applications of Dynamic Closest Pairs

David Eppstein
UC Irvine

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

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

General Terms: Closest Pair, Agglomerative Clustering

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

1. INTRODUCTION

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

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

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

Sequential recombination

k_t alg.: Find smallest of

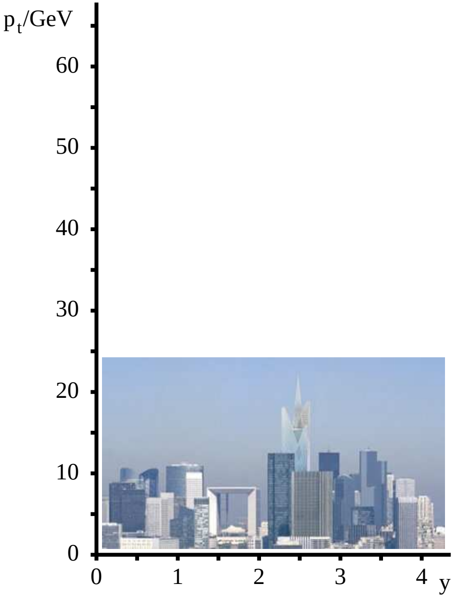
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



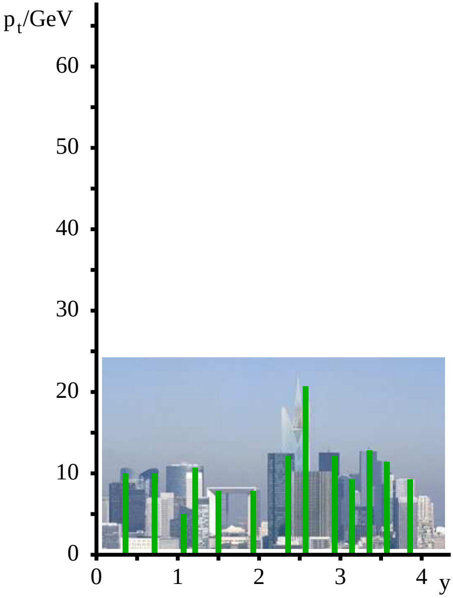
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

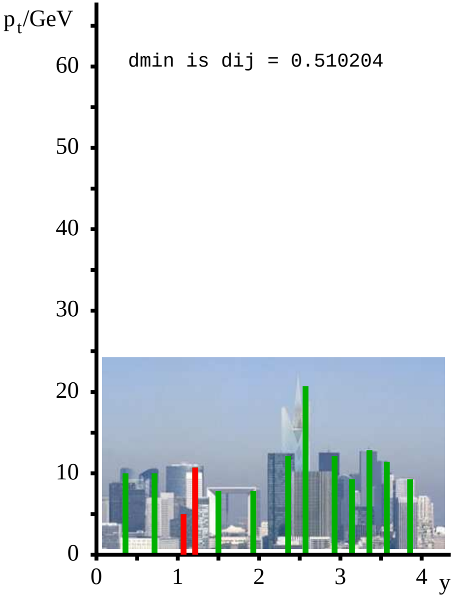
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

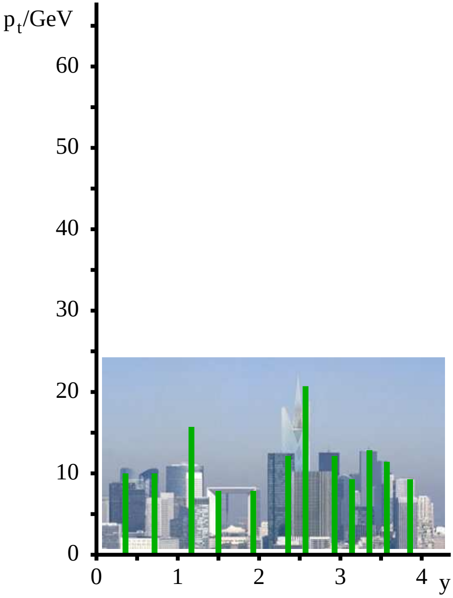
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



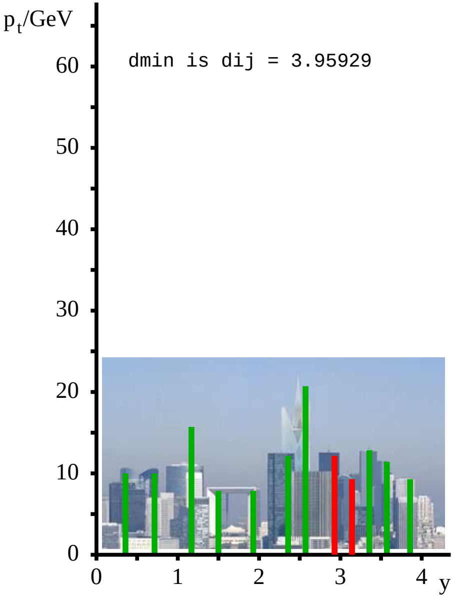
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

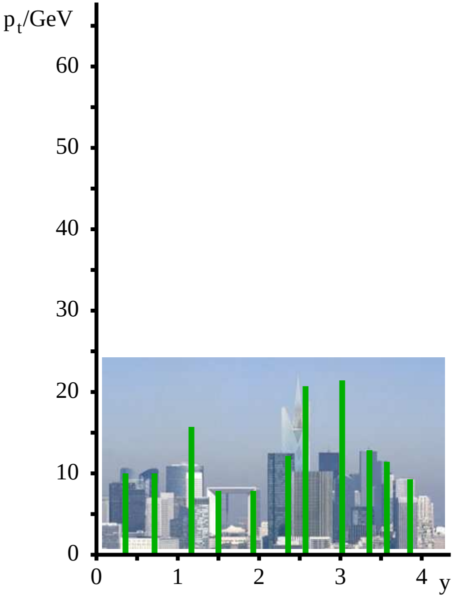
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

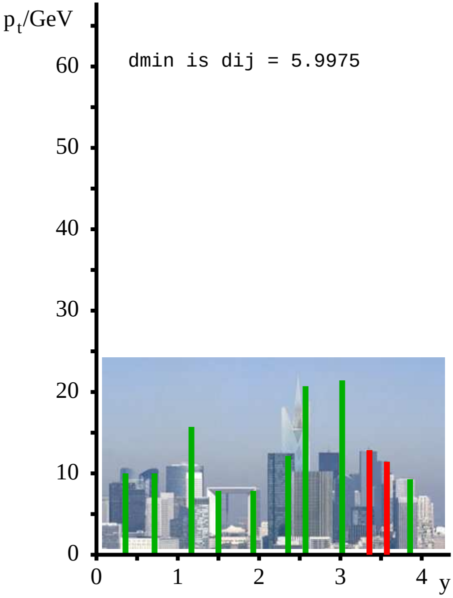
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

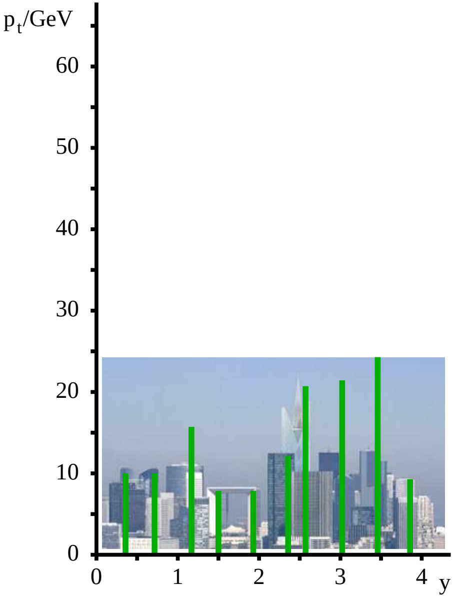
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

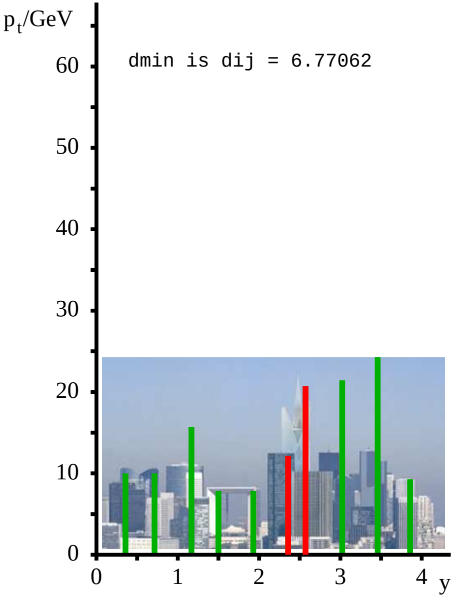
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



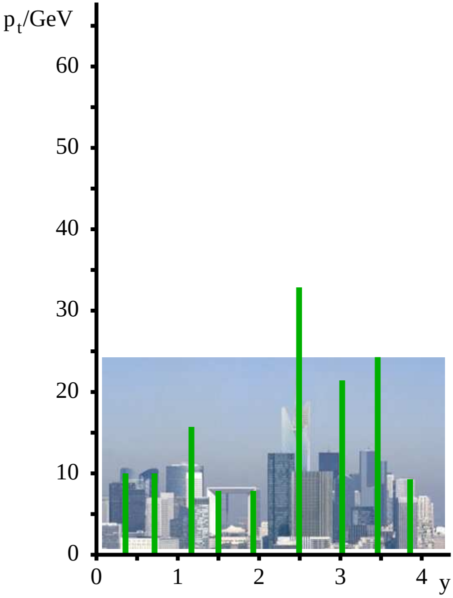
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



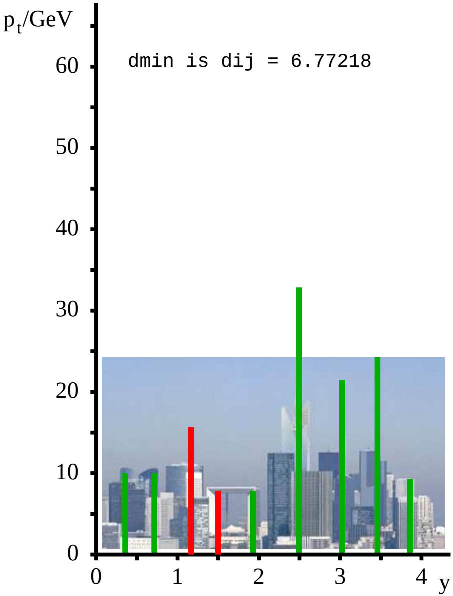
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

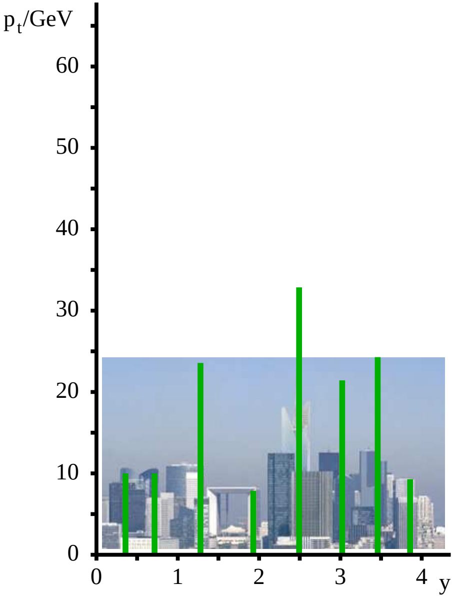
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



k_t alg.: Find smallest of

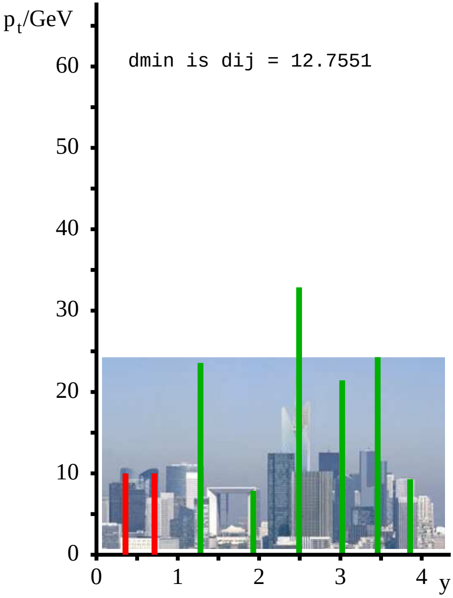
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

If d_{ij} recombine; if d_{iB}, i is a jet

Example clustering with k_t algorithm, $R = 0.7$

ϕ assumed 0 for all towers

Sequential recombination



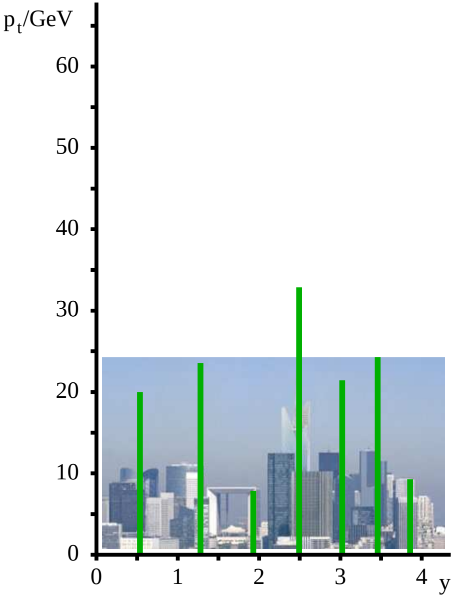
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



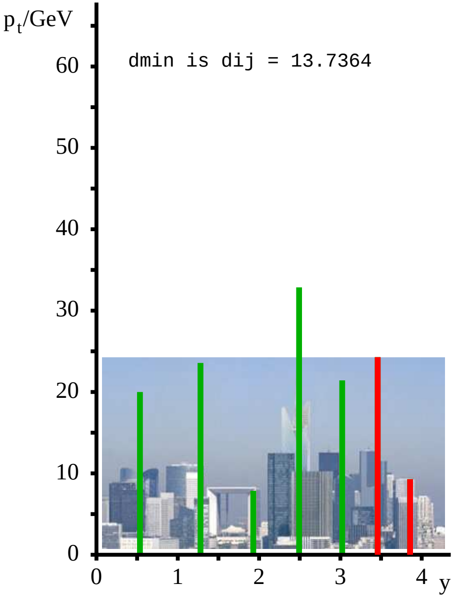
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



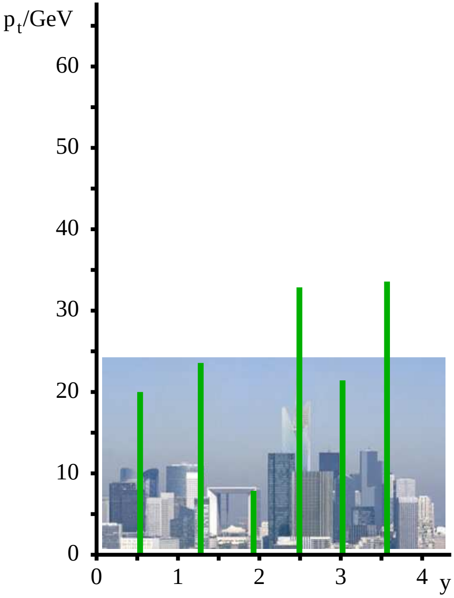
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



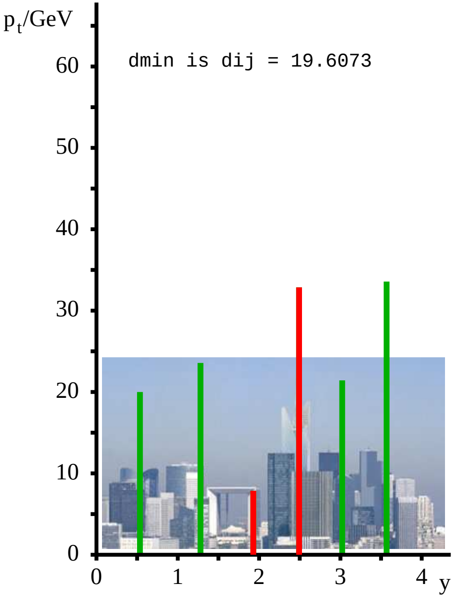
k_t alg.: Find smallest of

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

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

ϕ assumed 0 for all towers

Sequential recombination



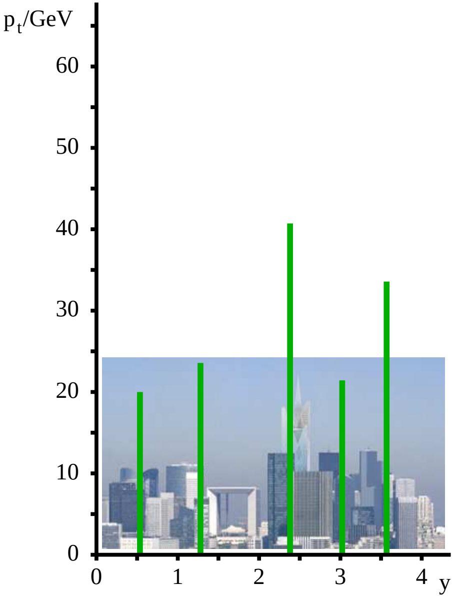
$\mathbf{k_t}$ alg.: Find smallest of

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

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

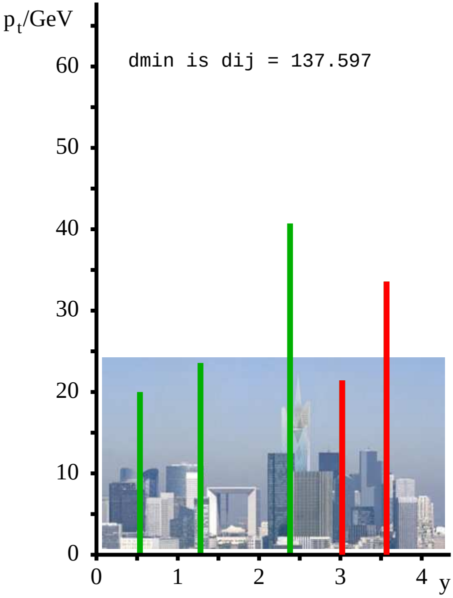
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

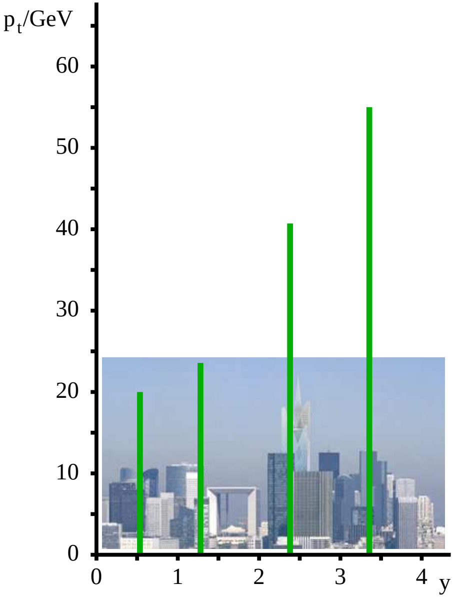
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

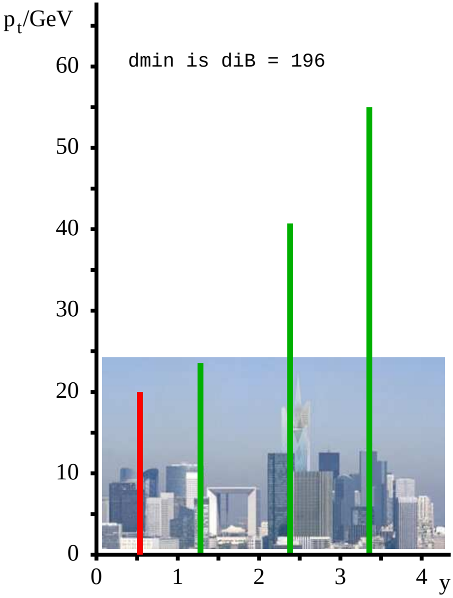
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



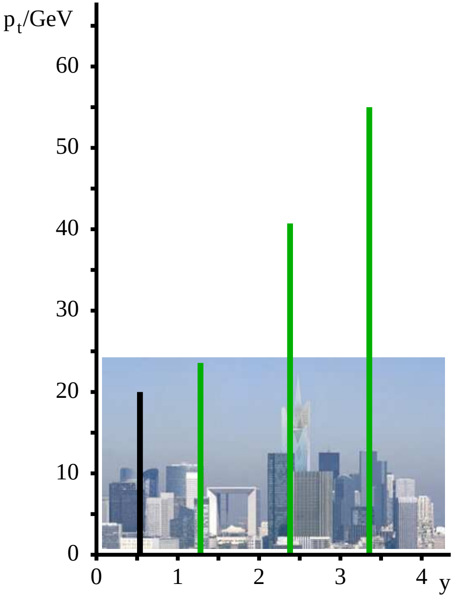
k_t alg.: Find smallest of

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

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

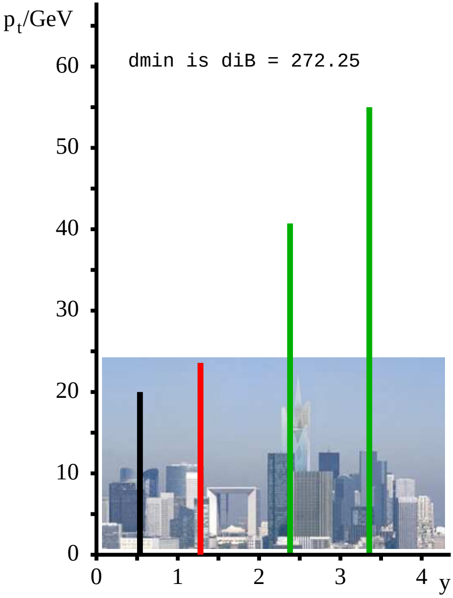
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

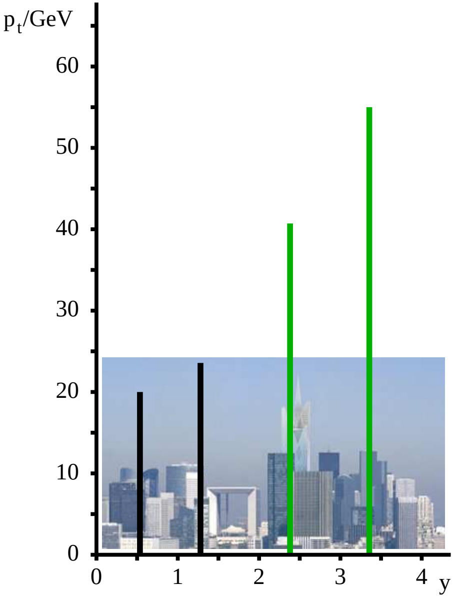
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

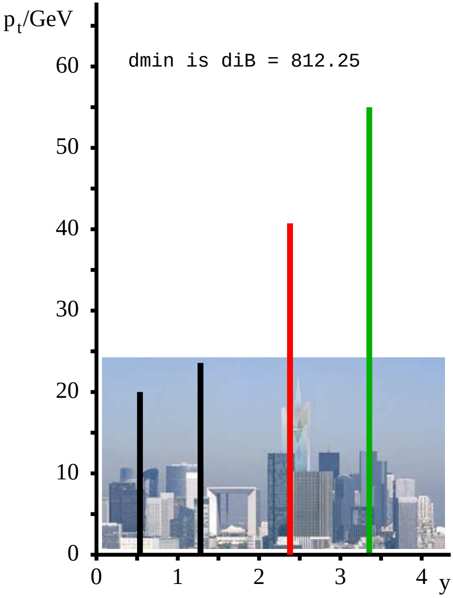
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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

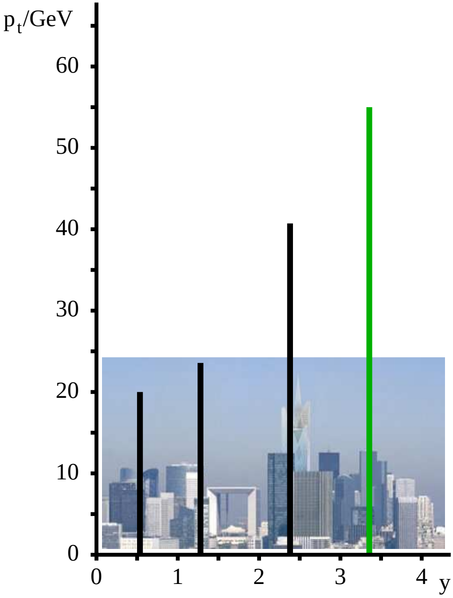
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



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

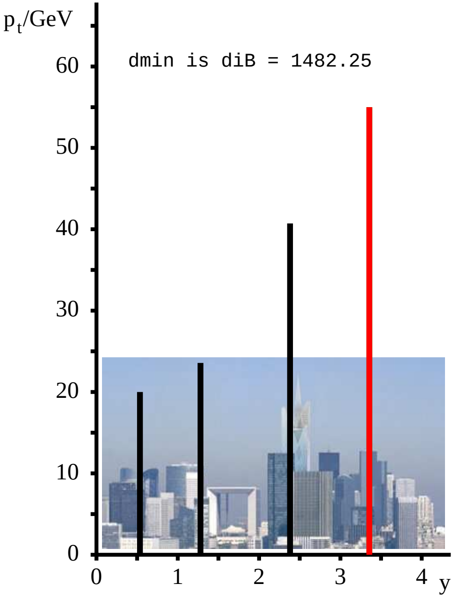
$$d_{ij} = \min(k_{ti}^2, k_{tj}^2)\Delta R_{ij}^2/R^2, \quad d_{iB} = k_{ti}^2$$

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



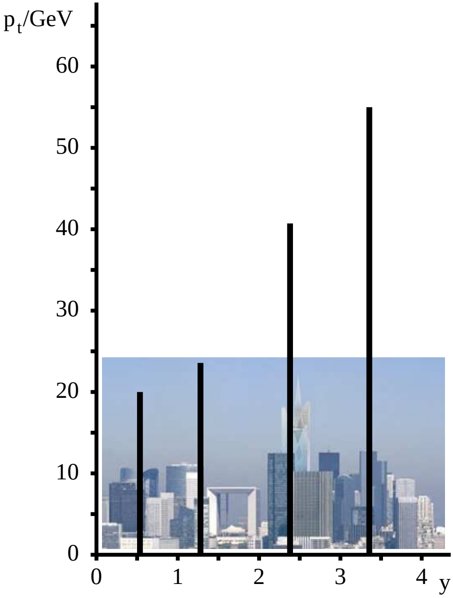
k_t alg.: Find smallest of

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

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

Cambridge/Aachen

Simplest of hadron-collider algorithms. Consider only angular divergences between particles:

1. Find pair of particles with smallest ΔR_{ij}
2. if $\Delta R_{ij} < R$ recombine them
3. otherwise stop: all remaining particles are the final jets

Dokshitzer, Leder, Moretti, Webber '97 (Cambridge): more involve e^+e^- form
had ordering in angle, soft-freezing in k_t distance

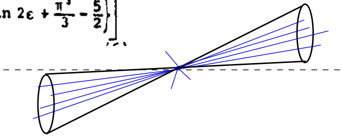
Wobisch & Wengler '99 (Aachen): simple inclusive hadron-collider form

Cone Origins

First ‘cone 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-

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



Cone basics

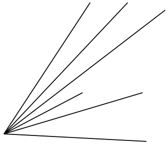
Modern cone algs have two main steps:

- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents
- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure

  [Blazey et al. '00 (Run II jet physics)]



Cone basics

Modern cone algs have two main steps:

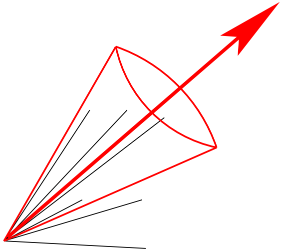
- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents

- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure

  [Blazey et al. '00 (Run II jet physics)]



Cone basics

Modern cone algs have two main steps:

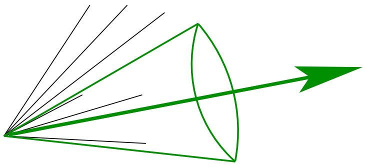
- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents

- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure

  [Blazey et al. '00 (Run II jet physics)]



Cone basics

Modern cone algs have two main steps:

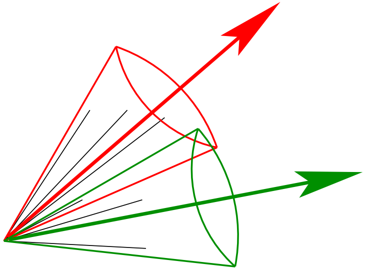
- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents

- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure

  [Blazey et al. '00 (Run II jet physics)]



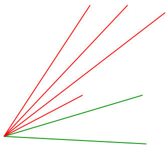
Cone basics

Modern cone algs have two main steps:

- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents
- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure
  [Blazey et al. '00 (Run II jet physics)]



Cone basics

Modern cone algs have two main steps:

- Find some/all stable cones

  ≡ cone pointing in same direction as the momentum of its contents

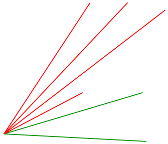
- Resolve cases of overlapping stable cones

  By running a 'split–merge' procedure
  [Blazey et al. '00 (Run II jet physics)]

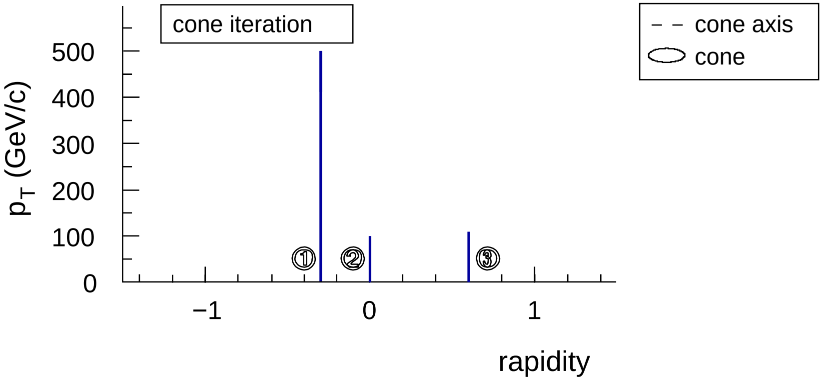
Qu: How do you find the stable cones?

Until recently used iterative methods:

- use each particle as a starting direction for cone; use sum of contents as new starting direction; repeat.

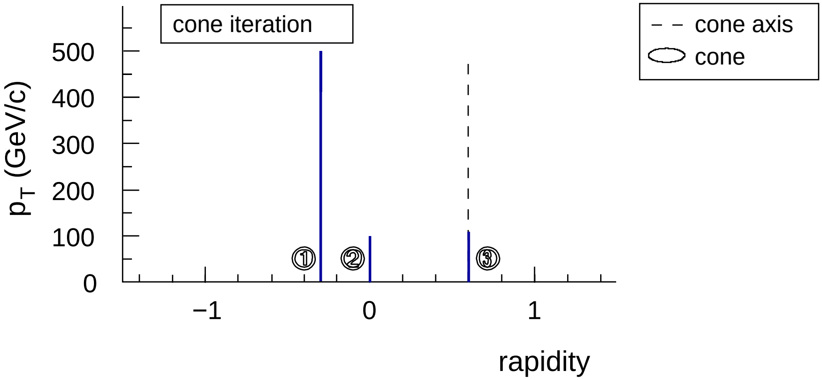


Iteration example



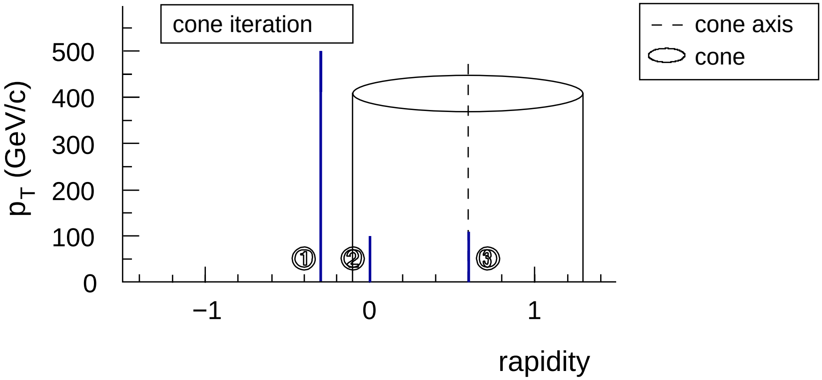
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



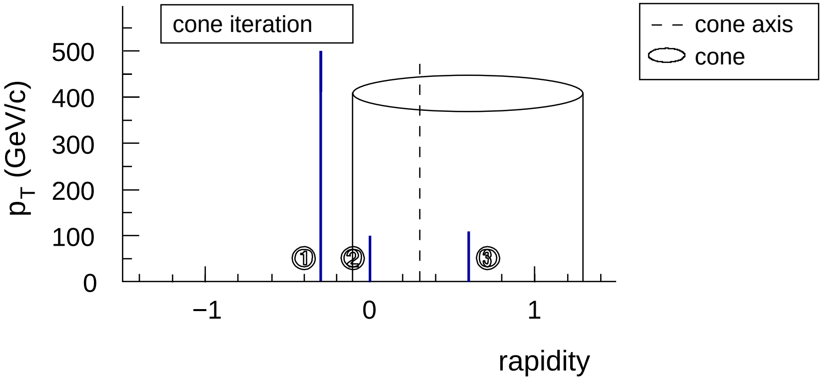
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



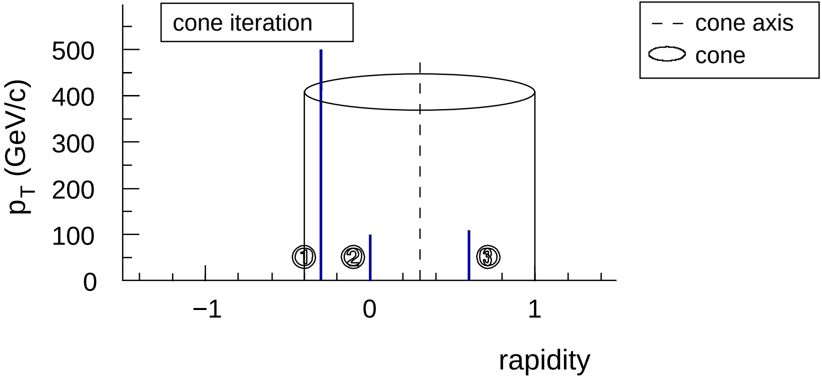
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



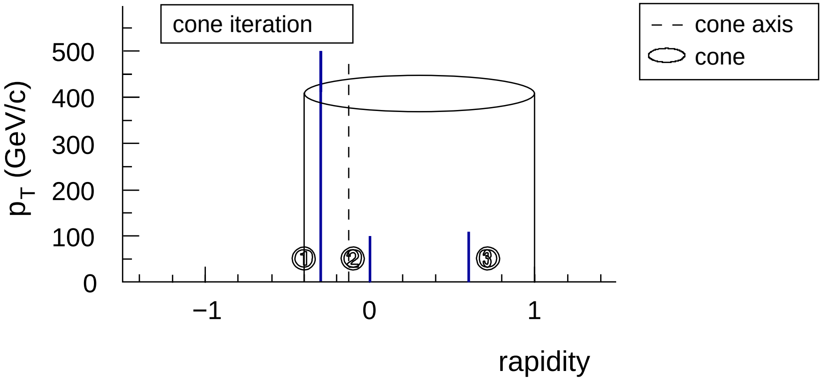
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



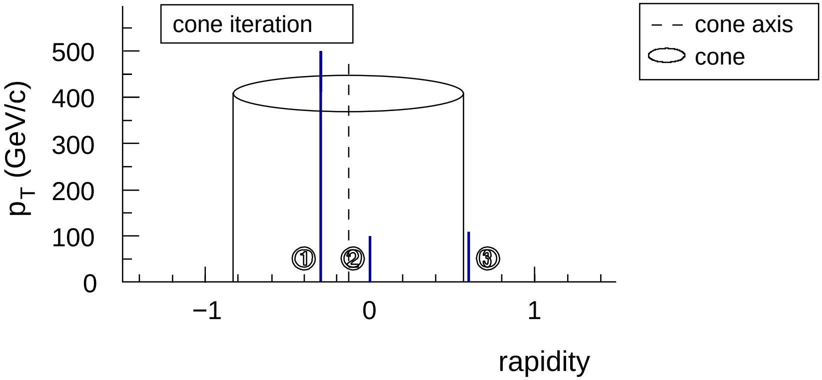
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



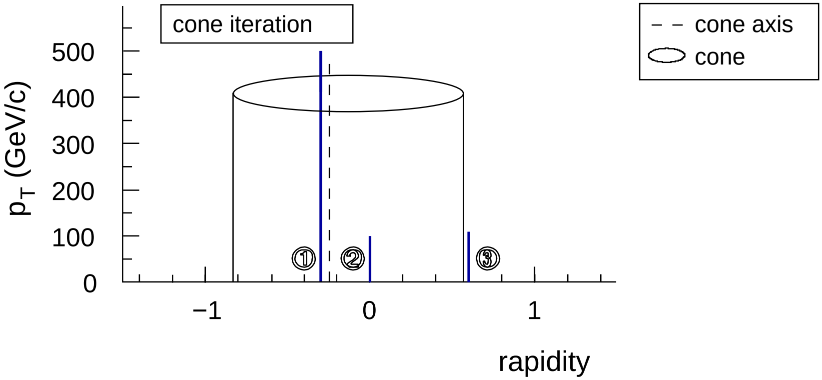
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



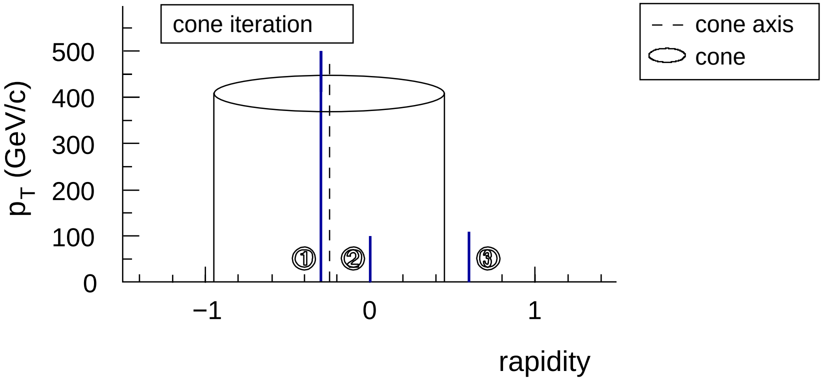
[These and related figures adapted/copied from Markus Wobisch]

Iteration example



[These and related figures adapted/copied from Markus Wobisch]

Iteration example



[These and related figures adapted/copied from Markus Wobisch]

Iterative cone problems

- What are the starting points for iteration?
- Start with hardest particle as seed: collinear unsafe
- Use all particles: extra soft one $\longrightarrow$ new solution

Iterative cone finding **plagued by IR and collinear unsafety problems**

Iterative cone problems

- What are the starting points for iteration?
- Start with hardest particle as seed: collinear unsafe
- Use all particles: extra soft one $\longrightarrow$ new solution

Iterative cone finding **plagued by IR and collinear unsafety problems**

Among consequences of IR unsafety:

	Last meaningful order	
	It. cone	MidPoint
Inclusive jets	LO	NLO
$W/Z + 1$ jet	LO	NLO
3 jets	**none**	LO
$W/Z + 2$ jets	**none**	LO
m_{jet} in $2j + X$	**none**	**none**

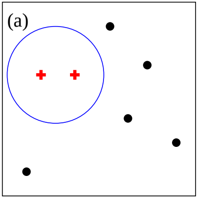
NB: \$30 – 50M investment in NLO

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

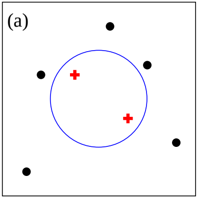
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

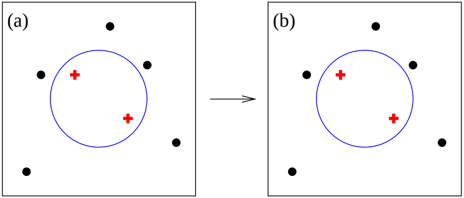
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

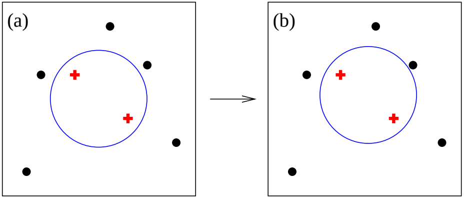
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

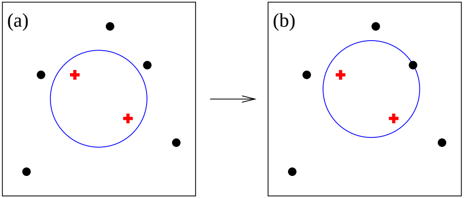
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

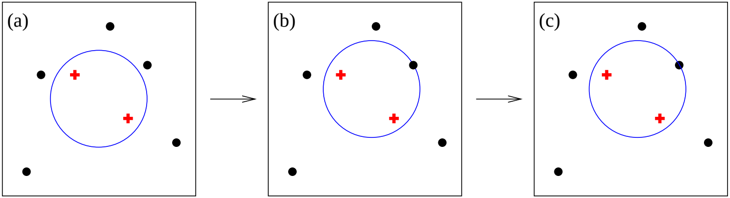
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

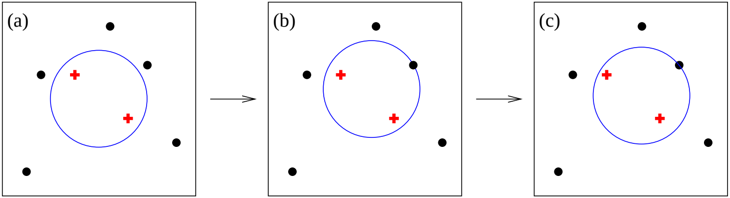
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

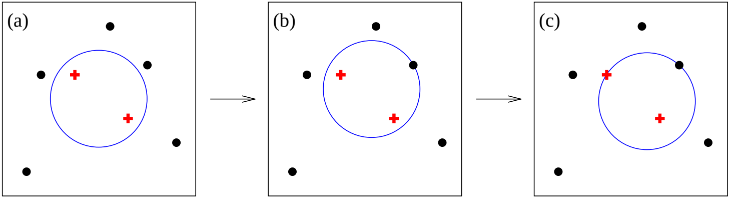
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

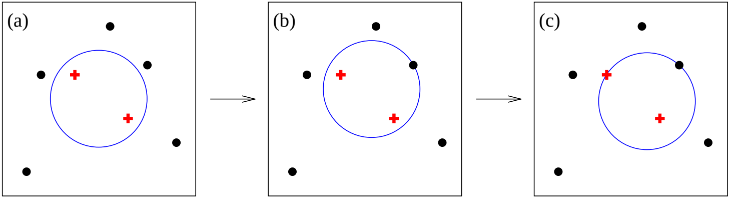
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

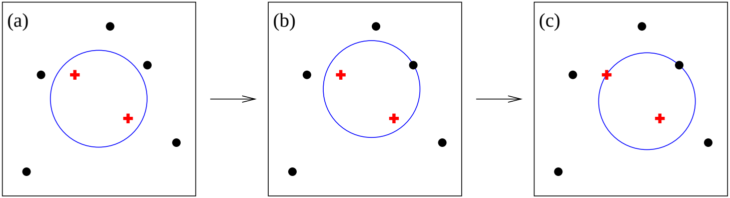
GPS & Soyez '07

Stable-cone finding: a geometrical problem

Cones are just *circles* in the $y - \phi$ plane. To find all stable cones:

1. Find all distinct ways of enclosing a subset of particles in a $y - \phi$ circle
2. Check, for each enclosure, if it corresponds to a stable cone

Finding all distinct circular enclosures of a set of points is *geometry:*



Any enclosure can be moved until a pair of points lies on its edge.

Result: Seedless Infrared Safe Cone algorithm (SISCone)

Runs in $N^2 \ln N$ time ($\simeq$ midpoint's N^3)

GPS & Soyez '07

SISCone algorithm as a whole

Put the set of current particles equal to the set of all particles in the event.

repeat

  Find *all* stable cones of radius R for the current set of particles, e.g. using algorithm 2.

  For each stable cone, create a protojet from the current particles contained in the cone, and add it to the list of protojets.

  Remove all particles that are in stable cones from the list of current particles.

until No new stable cones are found, or one has gone around the loop N_{pass} times.

Run a Tevatron Run-II type split–merge procedure, algorithm 3, on the full list of protojets, with overlap parameter f and transverse momentum threshold $p_{t,\text{min}}$.

SISCone part 2: finding stable cones

1: For any group of collinear particles, merge them into a single particle.
2: **for** particle $i = 1 \ldots N$ **do**
3: Find all particles j within a distance $2R$ of i. If there are no such particles, i forms a stable cone of its own.
4: Otherwise for each j identify the two circles for which i and j lie on the circumference. For each circle, compute the angle of its centre C relative to i, $\zeta = \arctan \frac{\Delta\phi_{iC}}{\Delta y_{iC}}$.
5: Sort the circles into increasing angle ζ.
6: Take the first circle in this order, and call it the current circle. Calculate the total momentum and checkxor for the cones that it defines. Consider all 4 permutations of edge points being included or excluded. Call these the "current cones".
7: **repeat**
8: **for** each of the 4 current cones **do**
9: If this cone has not yet been found, add it to the list of distinct cones.
10: If this cone has not yet been labelled as unstable, establish if the in/out status of the edge particles (with respect to the cone momentum axis) is the same as when defining the cone; if it is not, label the cone as unstable.
11: **end for**
12: Move to the next circle in order. It differs from the previous one either by a particle entering the circle, or one leaving the circle. Calculate the momentum for the new circle and corresponding new current cones by adding (or removing) the momentum of the particle that has entered (left); the checkxor can be updated by XORing with the label of that particle.
13: **until** all circles considered.
14: **end for**
15: **for** each of the cones not labelled as unstable **do**
16: Explicitly check its stability, and if it is stable, add it to the list of stable cones (protojets).
17: **end for**

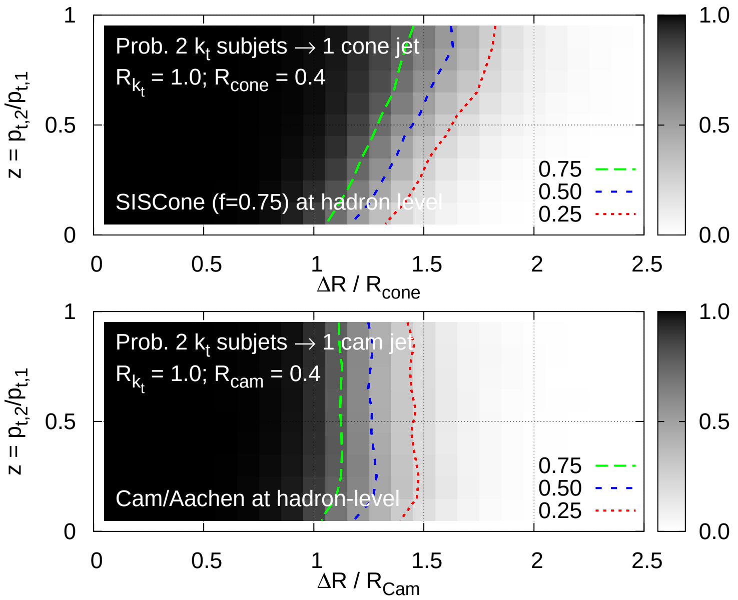
SISCone part 3: split–merge

1: **repeat**
Remove all protojets with $p_t < p_{t,\min}$.
Identify the protojet (i) with the highest $\tilde{p}_t$ ($\tilde{p}_{t,\mathrm{jet}} = \sum_{i \in \mathrm{jet}} |p_{t,i}|$).
Among the remaining protojets identify the one (j) with highest $\tilde{p}_t$ that shares particles (overlaps) with i.
5: **if** there is such an overlapping jet **then**
6: Determine the total $\tilde{p}_{t,\mathrm{shared}} = \sum_{k \in i \& j} |p_{t,k}|$ of the particles shared between i and j.
7: **if** $\tilde{p}_{t,\mathrm{shared}} < f \, \tilde{p}_{t,j}$ **then**
Each particle that is shared between the two protojets is assigned to the one to whose axis it is closest. The protojet momenta are then recalculated.
9: **else**
Merge the two protojets into a single new protojet (added to the list of protojets, while the two original ones are removed).
11: **end if**
12: If steps 7–11 produced a protojet that coincides with an existing one, maintain the new protojet as distinct from the existing copy(ies).
13: **else**
Add i to the list of final jets, and remove it from the list of protojets.
15: **end if**
16: **until** no protojets are left.

Sequential recombination v. cone jets

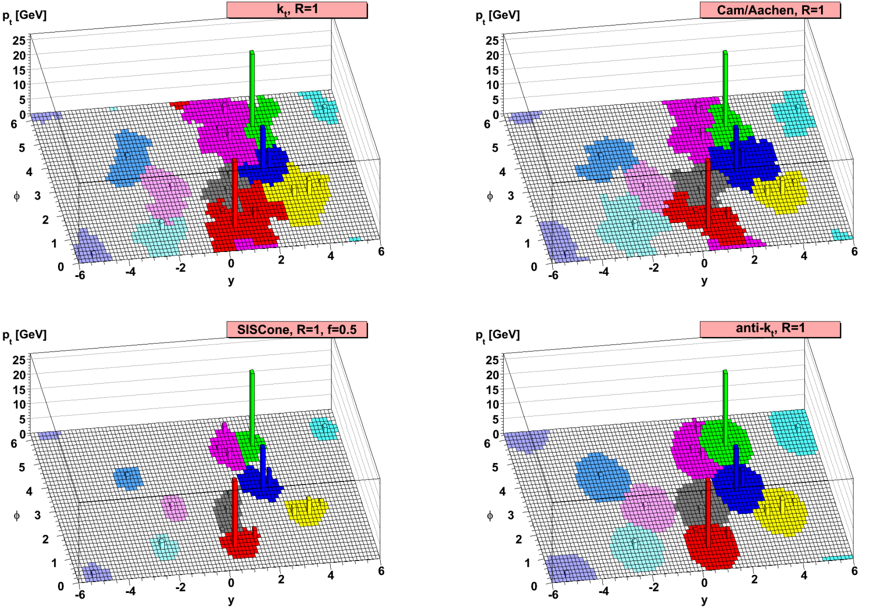
Sequential recombination	Cone
▶ simple	▶ complex
▶ gives you hierarchy	▶ no hierarchy
▶ two parameters: R, d_{cut}	▶ two parameters: R, f
▶ reaches equally for soft, hard particles	▶ reaches further for hard than soft particles
▶ jets with irregular boundaries	▶ less irregular boundaries
Loved by theorists e^+e^- experiments	Tolerated by theorists Most common in pp NB: many cones in existence All IRC unsafe except SISCone

Reach of jet algorithms



Herwig 6.510 + FastJet 2.1

Jet contours – visualised



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

Jets at work

LHC unprecedented from jet-finding point of view, in many respects:

- accuracies being sought (e.g. top mass)
- range of scales being probed
- complexity of final states (many jets)
- messiness of final states (underlying event, pileup)

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

Examples that follow: applying flexibility & advanced techniques in jet-finding.

Jets v. R

Parton $p_t \longrightarrow$ jet p_t

Ill-defined: MC "parton"

PT radiation:

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

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

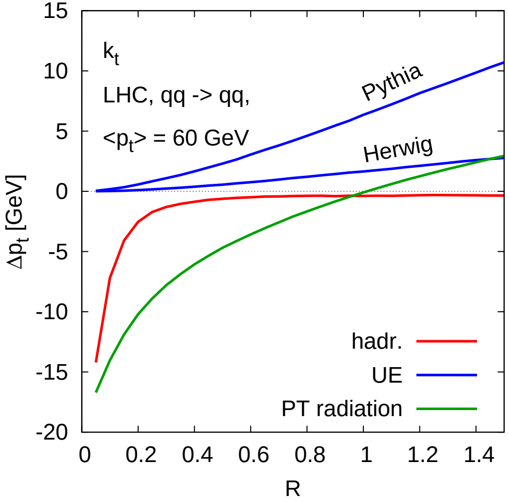
Hadronisation:

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

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

Underlying event:

$$q, g: \quad \Delta p_t \simeq \pi R^2 \cdot 0.5 - 2.5 \text{ GeV}$$



Cacciari, Dasgupta Magnea & GPS '07

Jets v. R

Parton $p_t \longrightarrow$ jet p_t

Ill-defined: MC "parton"

PT radiation:

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

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

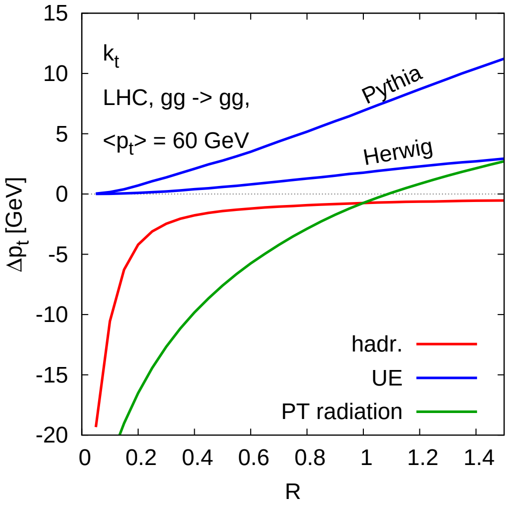
Hadronisation:

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

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

Underlying event:

$$q, g: \quad \Delta p_t \simeq \pi R^2 \cdot 0.5\text{–}2.5 \text{ GeV}$$



Cacciari, Dasgupta
Magnea & GPS '07

Jets v. R

Parton $p_t \longrightarrow$ jet p_t

Ill-defined: MC "parton"

PT radiation:

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

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

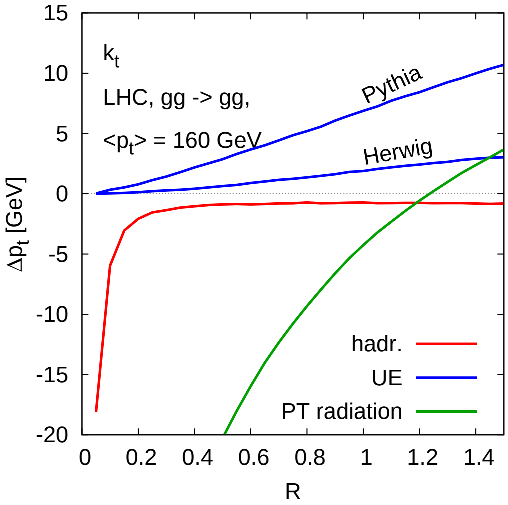
Hadronisation:

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

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

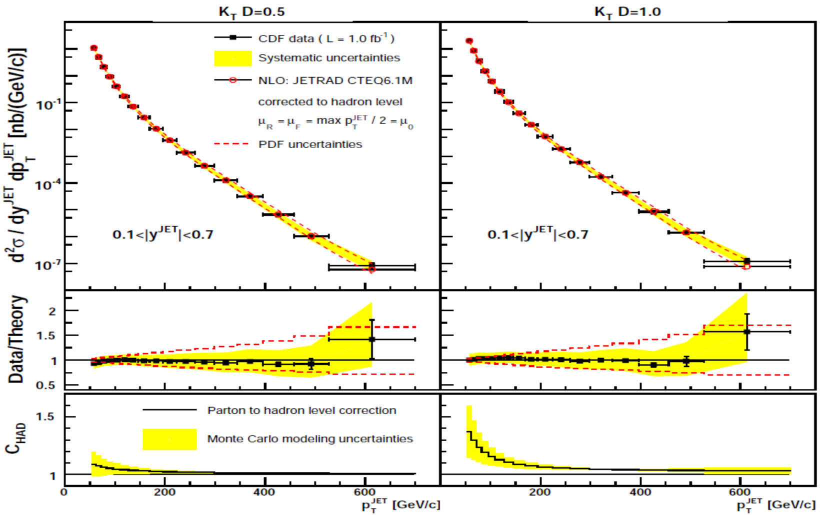
Underlying event:

$$q, g: \quad \Delta p_t \simeq \pi R^2 \cdot 0.5\text{—}2.5 \text{ GeV}$$



Cacciari, Dasgupta
Magnea & GPS '07

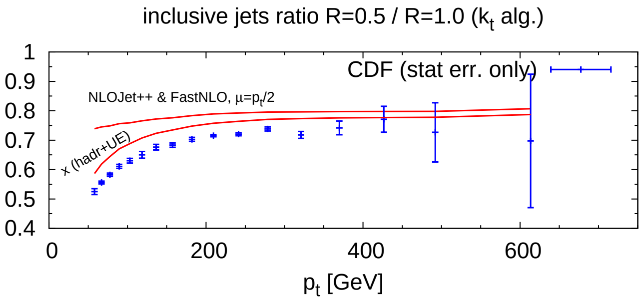
Inclusive jets: data v. theory



Agreement with theory independent of choice of R ($\equiv D$)

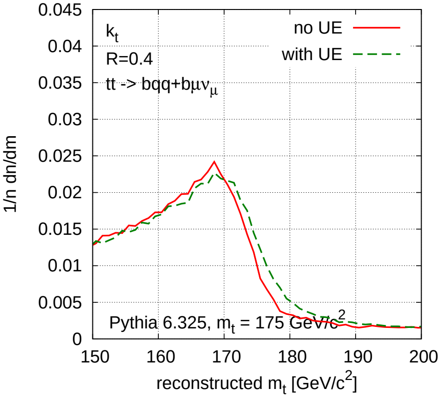
CDF, hep-ex/0701051

Inclusive jets: data v. theory



Agreement with theory independent of choice of R ($\equiv D$)
CDF, hep-ex/0701051

Robustness example



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

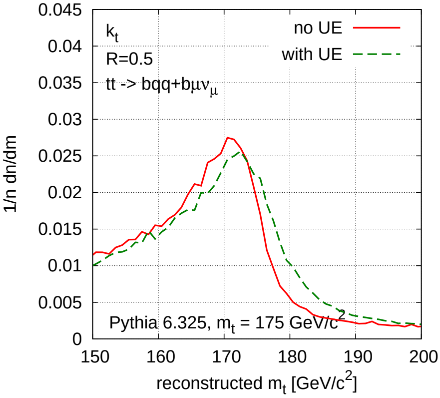
- Small *R*: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large *R*: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

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 example



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

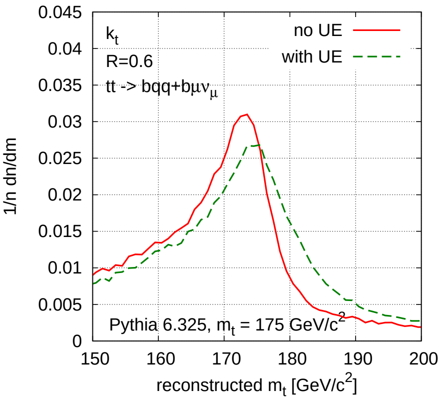
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2 - 4$ GeV.

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 example



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

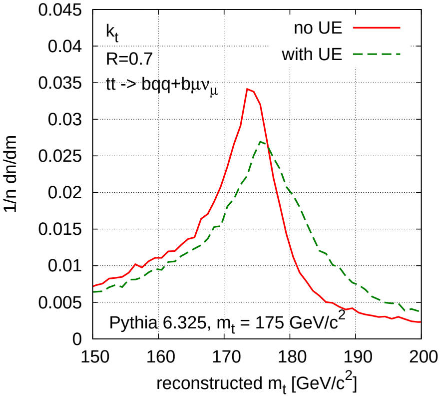
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

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 example



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

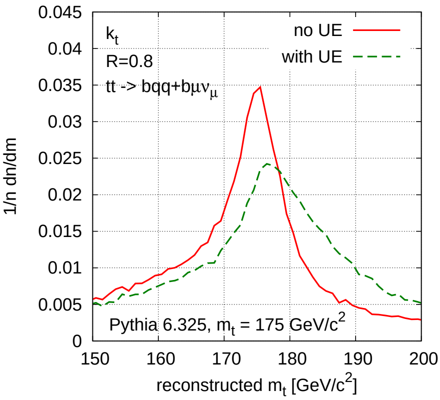
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

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 example



Game: measure top mass to 1 GeV

example for Tevatron

$m_t = 175$ GeV

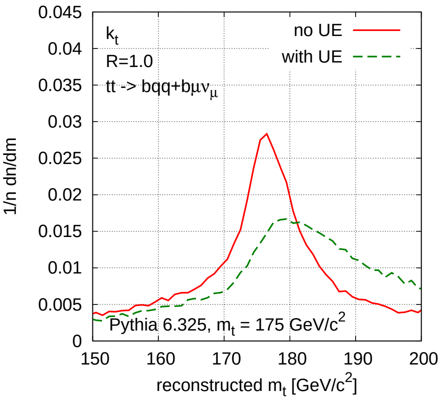
- Small R: lose 6 GeV to PT radiation and hadronisation, UE and pileup irrelevant
- Large R: hadronisation and PT radiation leave mass at ~ 175 GeV, UE adds $2-4$ GeV.

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 example



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.

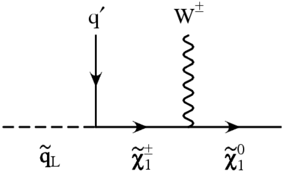
Is the final top mass (after W jet-energy-scale and Monte Carlo unfolding) independent of R used to measure jets?

Powerful cross-check of systematic effects

cf. Seymour & Tevlin '06

Multiscale problems

Boosted $W \to 2$ jets in SUSY decay chain



$$m_{\tilde{\chi}_1^{\pm}} \gg m_{\tilde{\chi}^0}, m_{W^{\pm}}$$

$$p_{tW} \gg m_W$$

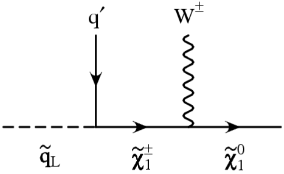
whole W in one jet

For same mass, signal and background have different distributions of $\sqrt{d_{ij}}$

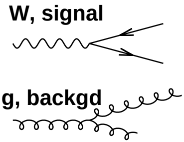
There's information **inside** jets

Multiscale problems

Boosted $W \to 2$ jets in SUSY decay chain



$$m_{\tilde{\chi}_1^{\pm}} \gg m_{\tilde{\chi}^0}, m_{W^{\pm}}$$



$$p_{tW} \gg m_W$$

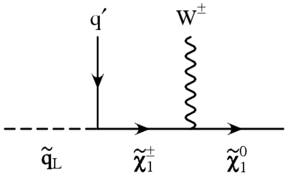
whole W in one jet

For same mass, signal and background have different distributions of $\sqrt{d_{ij}}$

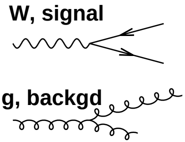
There's information **inside** jets

Multiscale problems

Boosted $W \to 2$ jets in SUSY decay chain



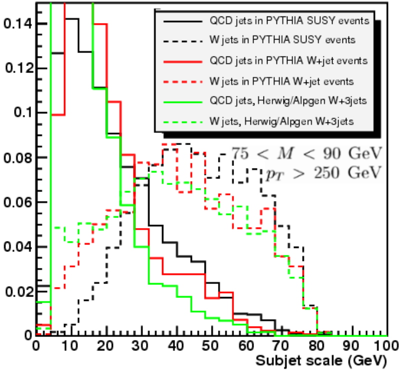
$$m_{\tilde{\chi}_1^\pm} \gg m_{\tilde{\chi}^0}, m_{W^\pm}$$



$p_{tW} \gg m_W$

whole W in one jet

For same mass, signal and background have different distributions of $\sqrt{d_{ij}}$

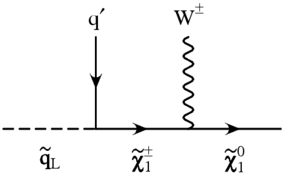


Butterworth, J.E. Ellis & Raklev '07

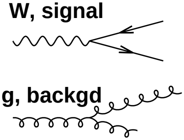
There's information **inside** jets

Multiscale problems

Boosted $W \to 2$ jets in SUSY decay chain



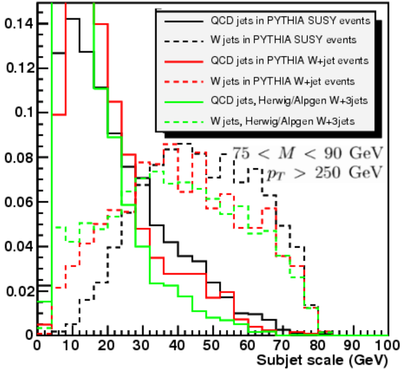
$$m_{\tilde{\chi}_1^\pm} \gg m_{\tilde{\chi}^0}, m_{W^\pm}$$



$$p_{tW} \gg m_W$$

whole W in one jet

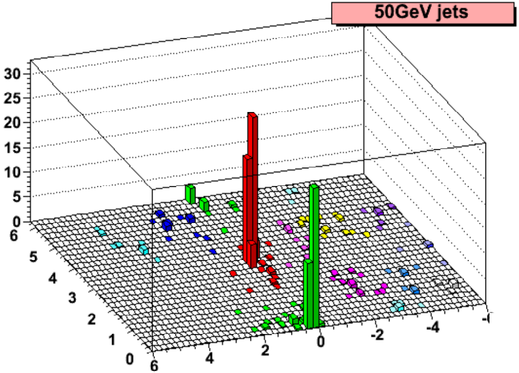
For same mass, signal and background have different distributions of $\sqrt{d_{ij}}$



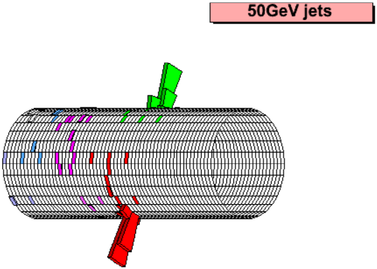
Butterworth, J.E. Ellis & Raklev '07

There's information inside jets

Jets, pileup and areas



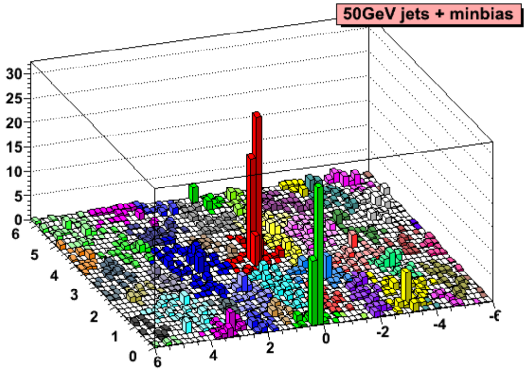
'Standard hard' event
Two well isolated jets



$\sim$ 200 particles

Easy even with old methods

Jets, pileup and areas

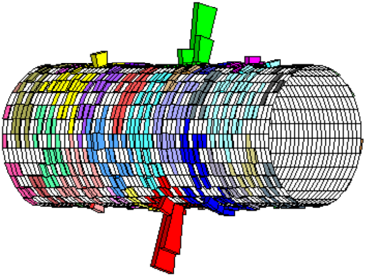


Add 10 min-bias events (moderately high lumi)

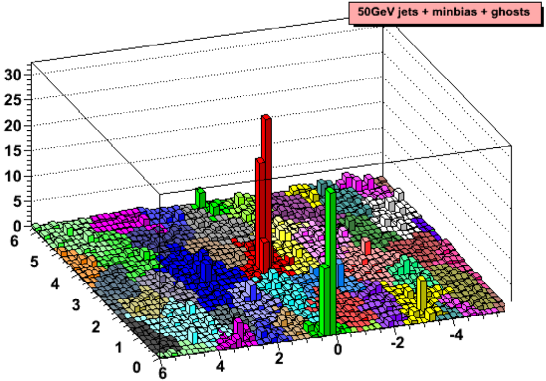
$\sim$ 2000 particles

Clustering takes $\mathcal{O}(10s)$ with old methods.

20ms with `FastJet`.



Jets, pileup and areas



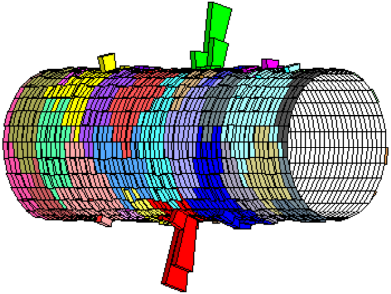
Add dense coverage of infinitely soft *"ghosts"*

See how many end up in jet to measure jet area

~ 10000 particles

Clustering takes ~ 20 minutes with old methods.

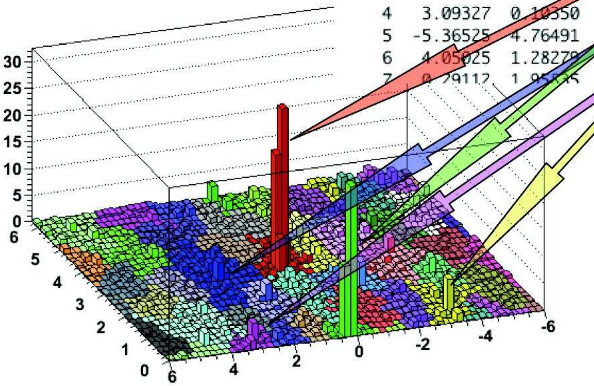
0.6s with `FastJet`.



Jet areas

```
iev 0 (irepeat 24): number of particles = 1428
strategy used =  NlnN
number of particles = 9051
Total area: 76.0265
Expected area: 76.0265
```

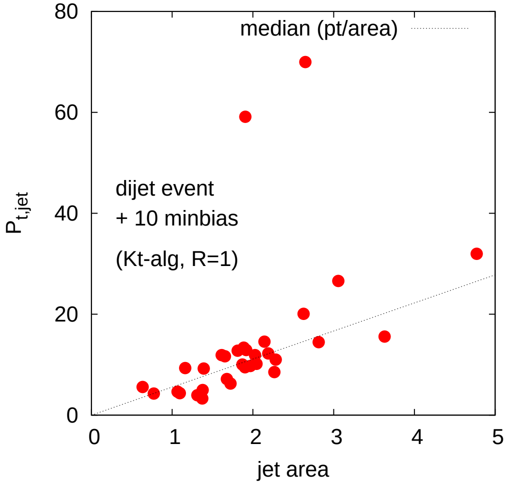
ijet	eta	phi	Pt	area	+-	err
0	0.15050	3.24498	69.970	2.625	+-	0.020
1	0.18579	0.13150	59.133	1.896	+-	0.020
2	2.33840	3.23960	31.976	4.749	+-	0.028
3	-3.41796	0.52394	26.595	3.084	+-	0.021
4	3.09327	0.10350	20.072	2.688	+-	0.023
5	-5.36525	4.76491	19.593	2.780	+-	0.012
6	4.05025	1.28279	15.561	3.592	+-	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

Jet areas



Jet areas in k_t algorithm are quite varied

Because k_t-alg adapts to the jet structure

- Contamination from min-bias $\sim$ area

Complicates corrections: min-bias subtraction is different for each jet.

Cone supposedly simpler
Area $= \pi R^2$?

Area-based subtraction

Basic Procedure:

- Use p_t/A from majority of jets (pileup jets) to get level, ρ, of pileup and UE in event
- Subtract pileup from hard jets:

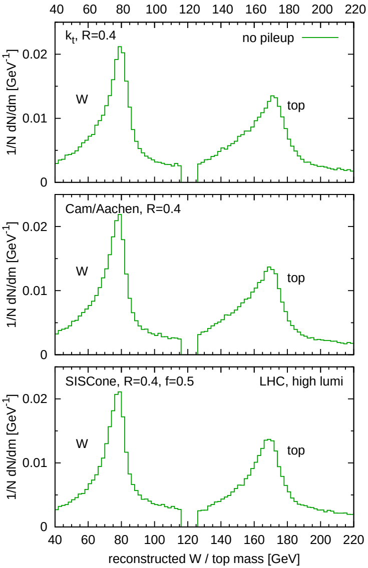
$$p_t \to p_{t,sub} = p_t - A\rho$$

Cacciari & GPS '07

Illustration:

- semi-leptonic $t\bar{t}$ production at LHC
- high-lumi pileup ($\sim$ 20 ev/bunch-X)

Same simple procedure works for a range of algorithms



Area-based subtraction

Basic Procedure:

- Use p_t/A from majority of jets (pileup jets) to get level, ρ, of pileup and UE in event
- Subtract pileup from hard jets:

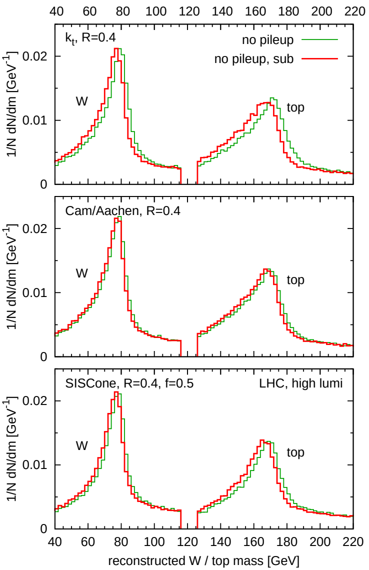
$$p_t \to p_{t,sub} = p_t - A\rho$$

Cacciari & GPS '07

Illustration:

- semi-leptonic $t\bar{t}$ production at LHC
- high-lumi pileup ($\sim$ 20 ev/bunch-X)

Same simple procedure works for a range of algorithms



Area-based subtraction

Basic Procedure:

- Use p_t/A from majority of jets (pileup jets) to get level, ρ, of pileup and UE in event
- Subtract pileup from hard jets:

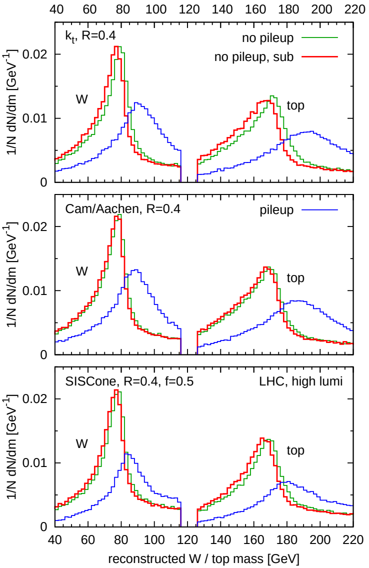
$$p_t \to p_{t,sub} = p_t - A\rho$$

Cacciari & GPS '07

Illustration:

- semi-leptonic $t\bar{t}$ production at LHC
- high-lumi pileup ($\sim$ 20 ev/bunch-X)

Same simple procedure works for a range of algorithms



Area-based subtraction

Basic Procedure:

- Use p_t/A from majority of jets (pileup jets) to get level, ρ, of pileup and UE in event
- Subtract pileup from hard jets:

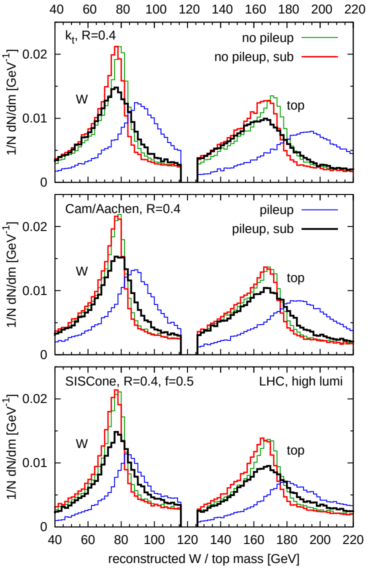
$$p_t \to p_{t,sub} = p_t - A\rho$$

Cacciari & GPS '07

Illustration:

- semi-leptonic $t\bar{t}$ production at LHC
- high-lumi pileup ($\sim$ 20 ev/bunch-X)

Same simple procedure works for a range of algorithms



Conclusions

- Know what algorithms you're using
- Be sure they're infrared & collinear safe

- Are your conclusions robust when you change algorithm?
- And when you vary R

- What are the scales in your problem
- Should you adapt your jet finding to the presence of disparate scales?

- As data arrives at LHC, new ideas in jet-definintion will arise, geared to actual LHC conditions. Keep your eyes open for best tools.

Conclusions

- Know what algorithms you're using
- Be sure they're infrared & collinear safe

- Are your conclusions robust when you change algorithm?
- And when you vary R

- What are the scales in your problem
- Should you adapt your jet finding to the presence of disparate scales?

- As data arrives at LHC, new ideas in jet-definintion will arise, geared to actual LHC conditions. Keep your eyes open for best tools.