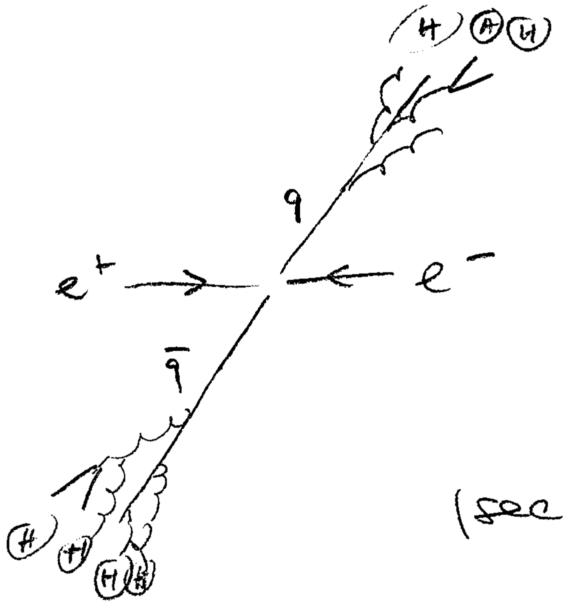
6.2 Jet algorithms

A disadvantage of event shapes is that they are not closely related to the underlying perturbative QCD dynamics. Collider events often have the form of jets, collimated sprays of hadrons which result after soft & collinear emissions and hadronisation



(see figures)

Jet algorithms identify such jets. In order for the jet rates to be perturbatively calculable, the algorithms should be IR safe.

There are two categories of jet algorithms

A.) <u>Cone-type</u> (Sterman-Weinberg, JetClu, Midpoint, ..., SISCone)

Cluster according to distance in coordinate space.

Put cones around dominant directions of energy flow.

B.) <u>Sequential</u> (k_T-algorithm, JADE, Cambridge/Aachen, ...)

Cluster according to distance in momentum space.

"Undo" the branchings in the perturbative evolution.

A thorough discussion of many algorithms can be found in 0906.1833 by Gavin Salam.

6.2.1 Sequential algorithms

JADE algorithm

1.) For each pair of particles calculate

$$y_{ij} = \frac{2E_iE_j(1-\cos\theta_{ij})}{Q} \left[= \frac{(p_i-p_j)^2}{Q} \text{ if } p_i^2=p_j^2=0 \right]$$

2.) Find minimum y_{min} of the y_{ij}'s

3.) If $y_{min} < y_{cut}$, then recombine i and j into new "particle" (or "pseudojet") and repeat from 1.)

4.) otherwise, declare all remaining particles to be jets and terminate.

The larger one chooses y_{cut}, the fewer jets one gets. The algorithm is infrared safe, because soft and collinear particles get clustered together immediately.

Disadvantage: soft particles get clustered together even if they move in opposite directions.

k_T - algorithm in e^+e^-

Identical to JADE, except

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

For $\theta_{ij} \to 0$ $\quad y_{ij} \to \min(E_i^2, E_j^2)\,\theta_{ij}^2$ .. transverse momentum

The use of the minimal energy ensures that soft particles are clustered with larger particles in similar directions.

k_T -, anti-k_T -, and C/A (Cambridge/Aachen) algorithm

At hadron colliders, the value of Q is not known, because the colliding quarks & gluons only carry a fraction of the proton momentum. For the same reason, the collisions are not happening in the c.m.s. Because of this one uses

Rapidity $\quad y = \frac{1}{2} \ln \left(\frac{E + p_z}{E - p_z} \right) \quad \left[\text{or } \eta = - \ln \tan \theta/2 \right]$

transverse momentum $\quad p_t = \sqrt{p_x^2 + p_y^2} \quad$ (beam in the z-dir)

azimuthal angle $\quad \phi = \arctan \frac{p_y}{p_x}$

as variables.

Rapidity is useful since rapidity differences are independent of boosts along the beam direction.

The distance measure in the above algorithms is

$$d_{ij} = \min \left(p_{t_i}^{2p}, p_{t_j}^{2p} \right) \frac{\Delta R_{ij}}{R^2}$$

$$d_{iB} = p_{ti}^{2p} \quad ; \quad \Delta R_{ij} = \sqrt{(y_i - y_j)^2 + (\phi_i - \phi_j)^2}$$

↖ "distance to beam"

$p=1$: k_t -algorithm (Catani et.al, Ellis and Soper '93)

$p=0$: C/A (Wobisch, Wengler '99)

$p=-1$: anti-k_t (Cacciari, Salam and Soyez '08)

The clustering is performed as follows

1.) Calculate all d_{iB}'s and d_{ij}'s and find the minimum.

2.) If it is a d_{ij}, combine i & j, goto 1.)

3.) If it is a d_{iB} declare it to be a jet and remove it from the list, goto 1.

4.) Stop when no particles remain.

Note that all particles are now part of a jet, since there is no d_{cut}

k_t was advocated by theorists because it is IR safe. It was disliked (and hardly used) by experimentalists because it produced irregular jets and was computationally expensive for many particles.

6.14.

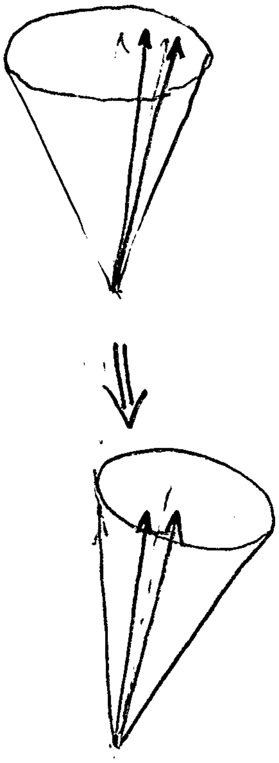
The irregular jets arise from combining soft particles early because their d_{ij}'s are low. This is improved by anti-k_t which combines hard particles first. It leads to regular jet-shapes, similar to cone algorithms and will presumably be the main algorithm used at the LHC.

6.2.2. Cone algorithms

We have already seen one example of a cone algorithm, the Sterman-Weinberg cross section.

Experimenters prefer cone algorithms, since they have a more immediate physical interpretation and are computationally less expensive (at least if one compromises on IR safety...)

Modern cone algorithms are iterative. One places a cone, calculates the total momentum of all particles inside the cone. Then one recenters the cone around this direction and repeats until one has a stable cone.



Two problems:

i.) What should be taken as the starting cones, the seeds of the iteration?

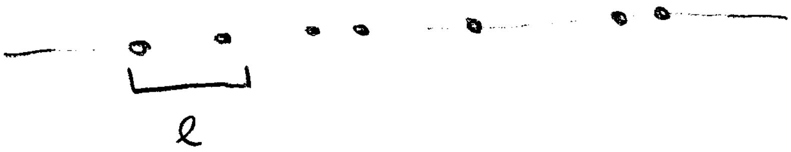
ii.) What should be done if two cones overlap?

The solution to i.) is to use a seedless cone algorithm: take all subsets of particles, calculate the total momentum, use it as the cone axis and see whether it corresponds to a stable cone. Since there are 2^N subsets, this is not feasible for a large number of particles.

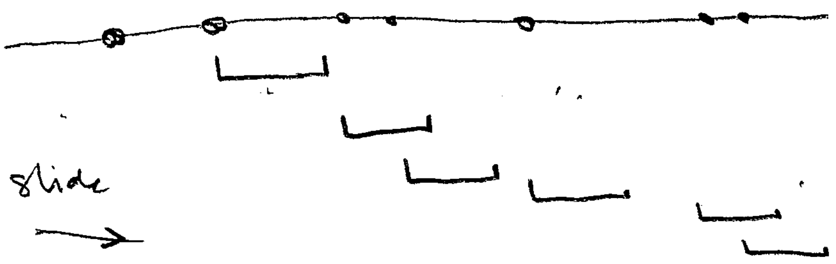
⇒ Experimentalists used seeds, which leads to IR unsafety, since additional soft particles can lead to new stable cones.

In '07 Salam and Soyez came up with a polynomial time seedless cone algorithm (SISCone)

Let's explain the procedure in 1-d:



A cone corresponds to a line of length l. To find all stable cones one can take all subsets of points and check wheter they are within a distance l. This is <u>very</u> inefficient. A beter algorithm is to slide a segment of length l along the line



In 2-d: (ϕ, y) one moves around a circle (see figures).

A solution to ii) is to use a so-called split-merge procedure

1.) Find the stable cone with the largest p_t, call it a.

2.) Find the next largest cone that shares particles, label it b. If no such cone exists, remove cone a and its particles, add a to the final list of jets.

3.) Calculate the p_t of the shared particles $p_{t,\text{shared}}$

a.) If $p_{t,\text{shared}} / p_{t,b} > f$, replace a & b with a single cone containing all particles

b.) otherwise split the two by assigning the shared particles to one of the jets.

4.) Repeat 1, until no cones are left.

6.19.

The program FastJet implments many algorithms. One can feed it a list of particle momenta, choose an algorithm and it spits out the jets.