Roma, June 2012

Clustering and jets

Matteo Cacciari
LPTHE Paris

Outline

- What is 'clustering'
  - Terminology, algorithms
  - Examples of applications: genetics, information retrieval
- Clustering in physics: jets
  - Role of jets
  - Examples of algorithms
  - Recent developments
    - Fast clustering
    - Jet areas
    - Jet substructure

Clustering v. Classification

Classification is **supervised** learning.

The classes need to have been defined in advance.
Then, you assign objects to them (after having been trained to do so).

Clustering is **unsupervised** learning.

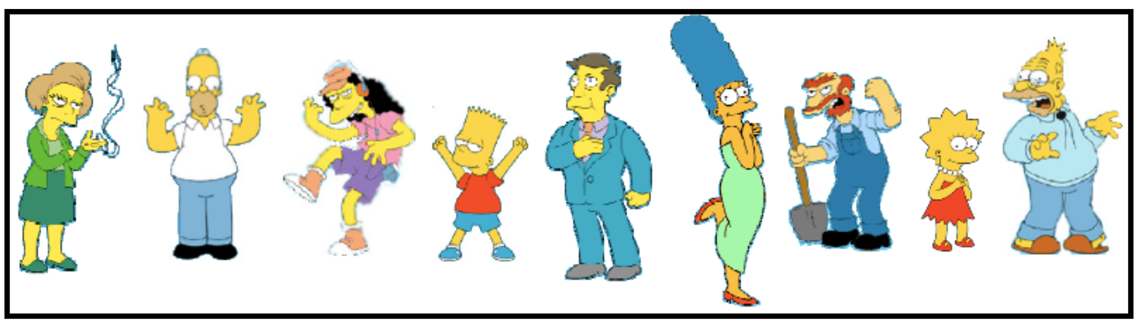
Performing the clustering is an unsupervised task. We do not know in advance what the results will be (e.g. what kind of clusters we'll find).
Rather, we want to learn a classification from the data.

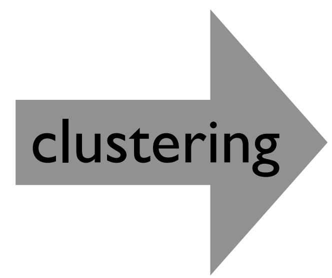
[Of course, this may only true up to a certain extent, because the definition of the similarity measure may say a lot about what we may be trying to achieve.]

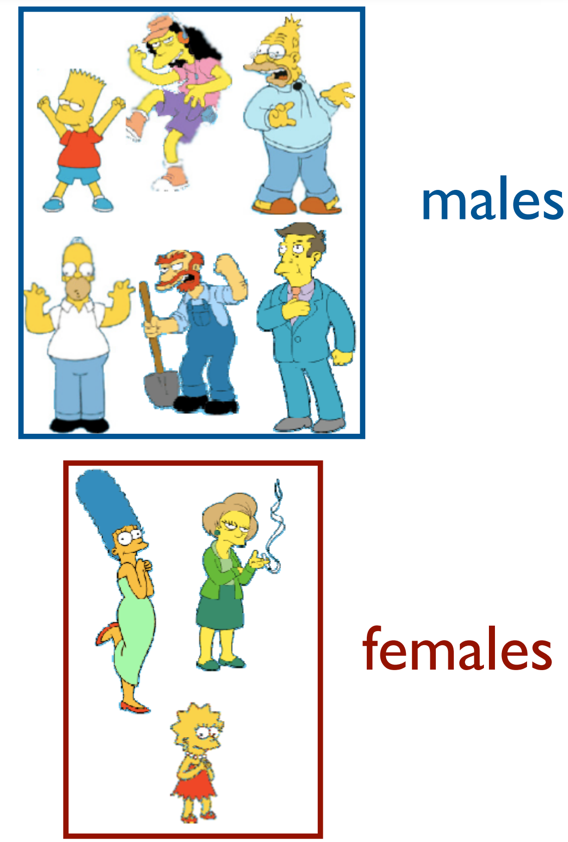
What is clustering?

Subdivide a set of items such that there is:

- **high intra**-cluster similarity
- **low inter-**cluster similarity







[Example by H. Lin]

NB. Of course, this should not be achieved simply by assigning elements to "male" or "female" classes, since it would be classification and not clustering.

Two big questions:

- What does **similar** mean?
- How to achieve the clustering in practice?

1) We need to specify what we mean when we say that two objects are similar

Definition of a **dissimilarity** between objects

2) We need to specify how we construct the clusters

Definition of a **clustering algorithm**

The combination of all possible dissimilarity measures with the many different clustering algorithms leads to an almost infinite number of combinations (not counting the different fields of application)

Distances (or dissimilarities)

The definition of 'distance' (or '**dissimilarity**') D(A,B) between two objects A and B is largely arbitrary, and can vary a lot depending on the objects (and therefore the context)

Almost infinite freedom in choosing the distance, as long as

$D(A,B) = D(B,A)$	(symmetry)
$D(A,B) = 0$ iff $A=B$	(self-similarity)
$D(A,B) \geq 0$	(positivity)
$D(A,B) \leq D(A,C) + D(B,C)$	(triangle inequality)

Of course, the quality of the results will depend on the choice
(and some specific contexts, like physics, can have further requirements)

The result of a clustering is only as good as the choice of the dissimilarity function used (but it can still be sub-optimal if a bad algorithm is used)

Linkage method

While the dissimilarity function D(A,B) gives the distance between two 'fundamental' objects, the distance between **clusters of objects** can still be defined in different ways.

Non-exhaustive list

'Single link': cluster distance = distance of two **closest** members

Gives long, thin clusters ('chain effect')

'Complete link': cluster distance = distance of two **most distant** members

Gives tight, roundish clusters

'Average link': cluster distance = average of all distances of all members

Robust against noise

'Centroid link': cluster distance = distance of centroids of clusters

Liked by physicists

The results of the algorithm can depend quite significantly on the linkage type used.
→ Yet another degree of freedom in constructing different clustering algorithm.

Clustering algorithms

Partitional: construct partitions of the set of objects and evaluate them by some criterion.

X Number of clusters must be provided in advance

✓ Easy to implement, fast

Hierarchical: create a hierarchical decomposition of the set of objects by some criterion

X Computationally heavy

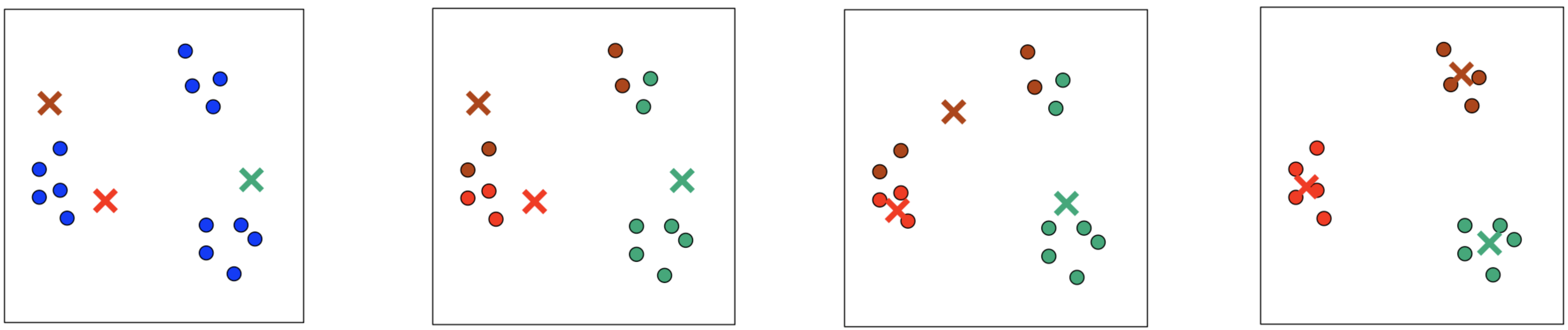
✓ No need to specify number of clusters in advance

✓ 'In depth' view of structure (hierarchy).
Can decide a posteriori on a criterion for number of clusters.

k-means

Example of a partitional algorithm

1) Choose K centroids at random
2) Assign objects to closest centroid, forming K clusters
3) Calculate centroid (mean of distances) of each cluster, update centroids
4) Check if an object in a cluster is closer to another centroid.
   Reallocate in case.
5) Repeat from step 3 until no object changes cluster anymore.



Step 1 (random centroids) | Step 2 (allocate objects) | Step 3 (move centroids) | Step 5 (end of iteration)

[Figures by E. Garrett-Mayer]

One of the main shortcomings:

result of final convergence can be highly sensitive to choice of initial seeds.

Also, the concept of 'mean distance' (to calculate the centroid) must be defined.

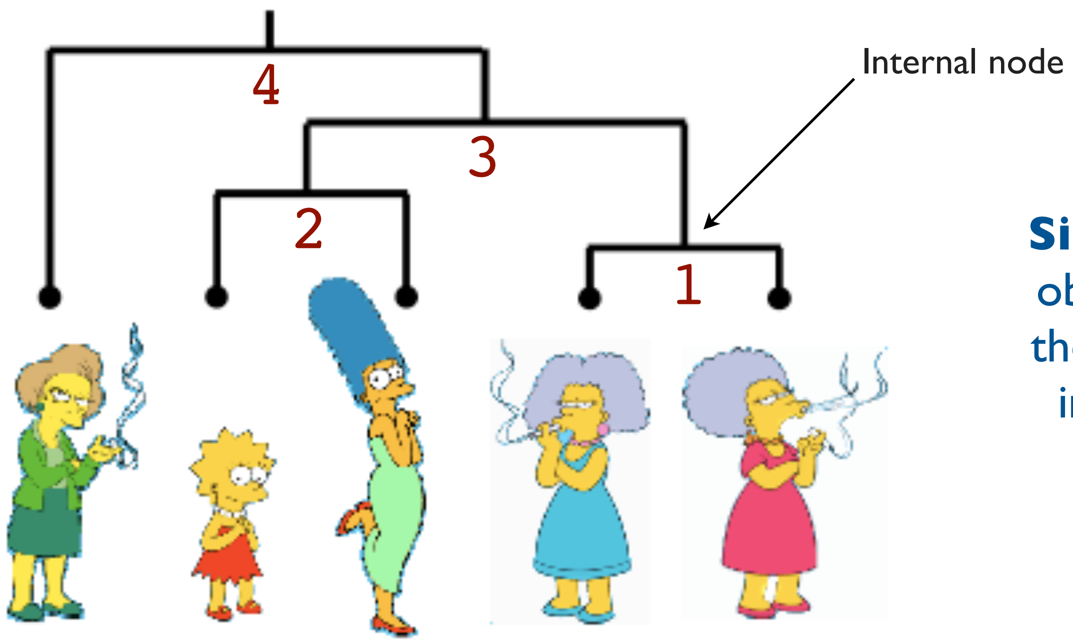
Agglomerative clustering

Example of a hierarchical algorithm

(often denoted HAC: Hierarchical Agglomerative Clustering)

1) Choose a dissimilarity function, calculate distance matrix between all objects
2) Choose a linkage method
3) **Cluster** two objects with **smallest** dissimilarity
4) Update the distance matrix
5) Repeat from step 3 until a single cluster is left
6) Look at the resulting hierarchy, and decide what 'best' number of clusters is

Dendrogram

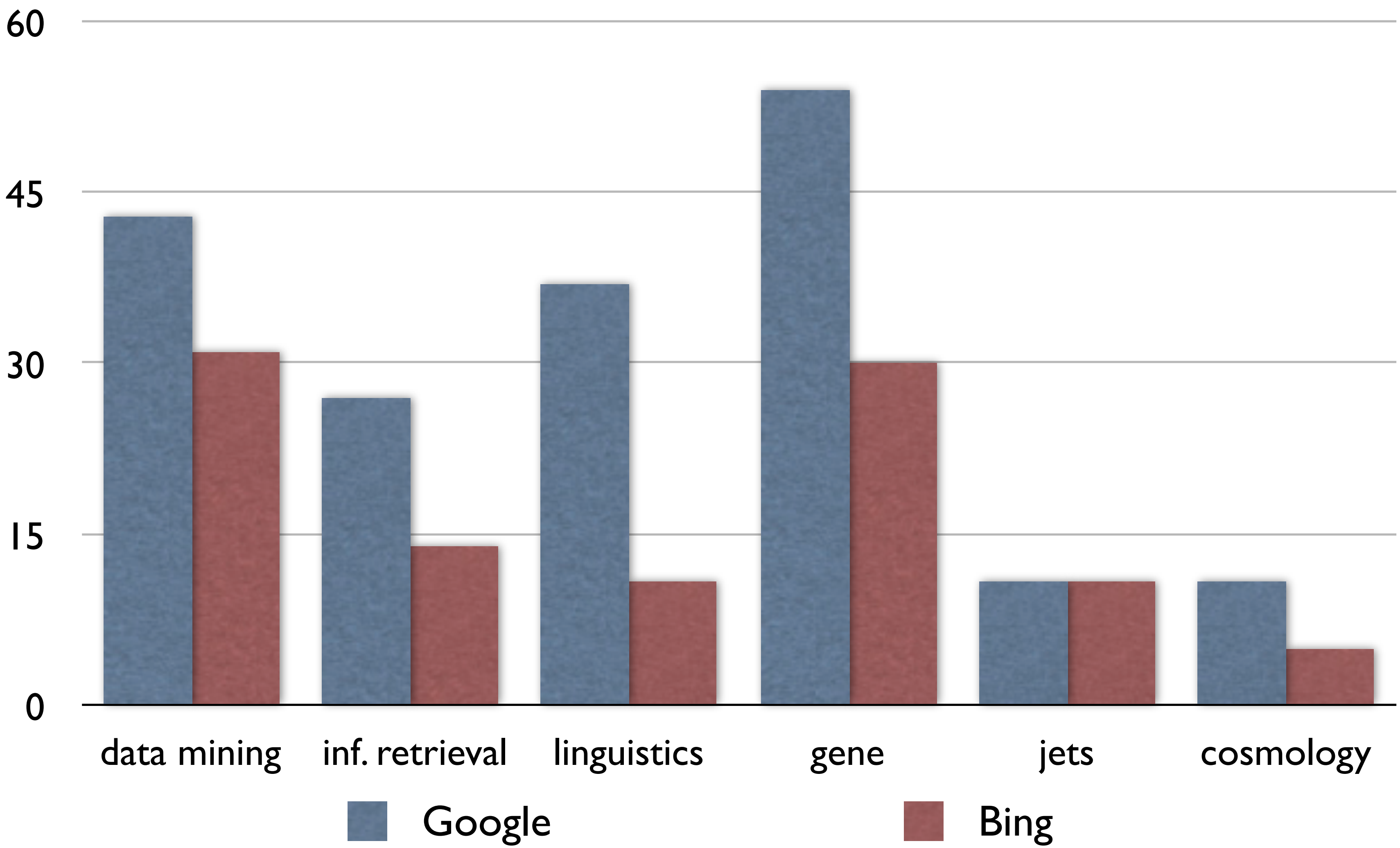


Similarity between two objects is represented by the **height** of the lowest internal node that they share.

Order of clustering is 1,2,3,4

Who uses clustering?

$$\frac{\text{\# documents containing ``clustering algorithm'' and X}}{\text{\# documents containing ``clustering algorithm''}} \quad (\times\ 100)$$



Distances (or dissimilarities)

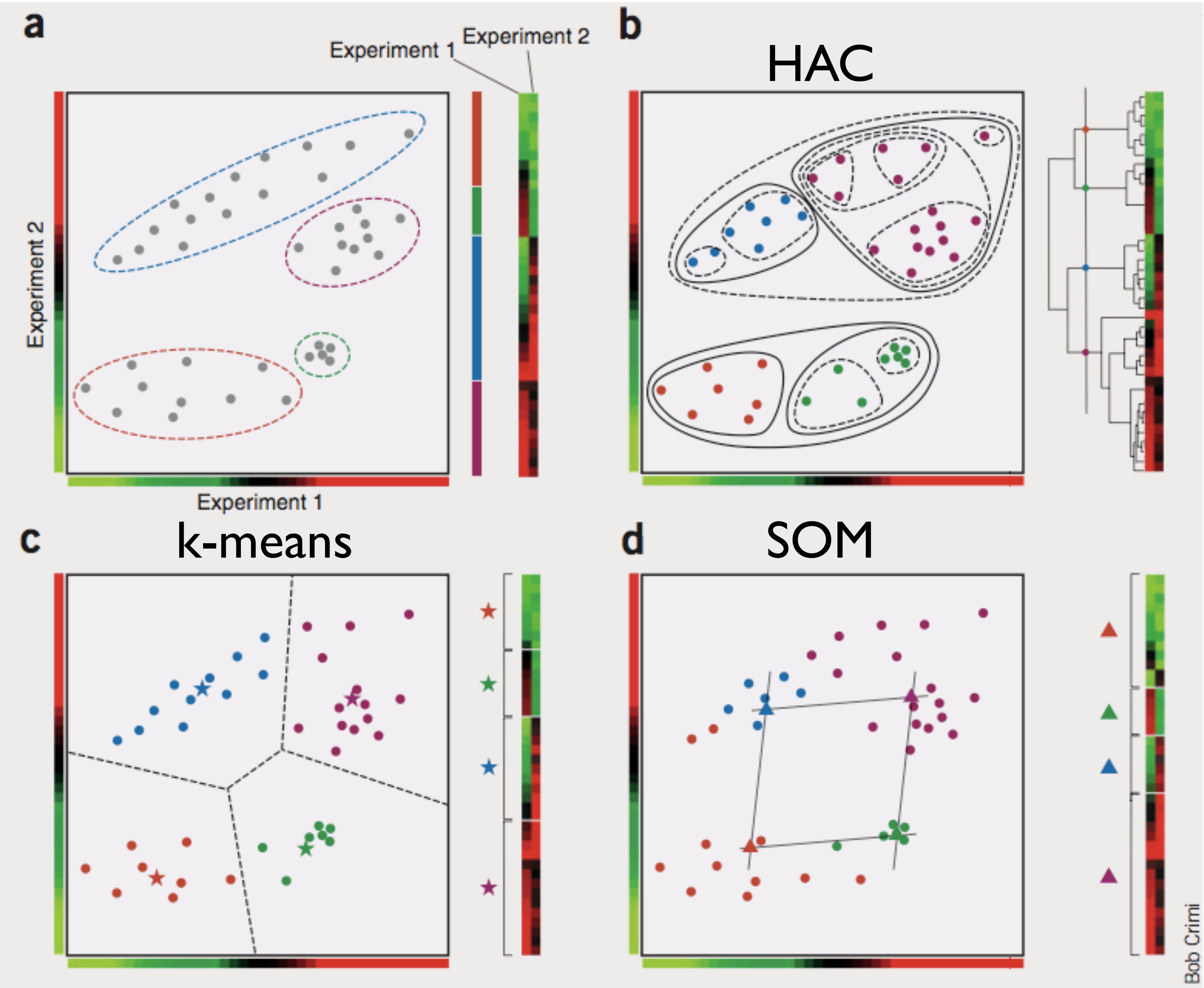
Examples of ingredients used to build ‘distances’

- **genetics**: correlation coefficients of expression values of genes in samples
- **information retrieval**: frequency of words in documents
- **physics**: direction (and possibly magnitude) of a particle’s 4-momentum

Each domain has its own specificities, and multiple choices or variations are usually available even within a specific context

Clustering and genes

Clustering used on gene expression data for '**class discovery**', i.e. find out which genes work together



[P. D'haeseleer, Nature vol. 23, n.12 (2005)]

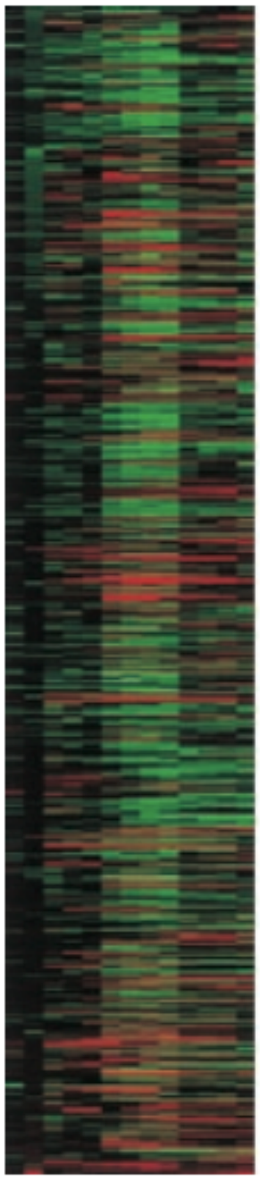
Gene expression 'heatmap' in **two** dimensions (i.e. experiments, or samples)

Rows are **samples**, columns are **genes**, colour is (some function of) gene **activity**

In two dimensions it's easy to 'visualize' clusters. What about with many samples?

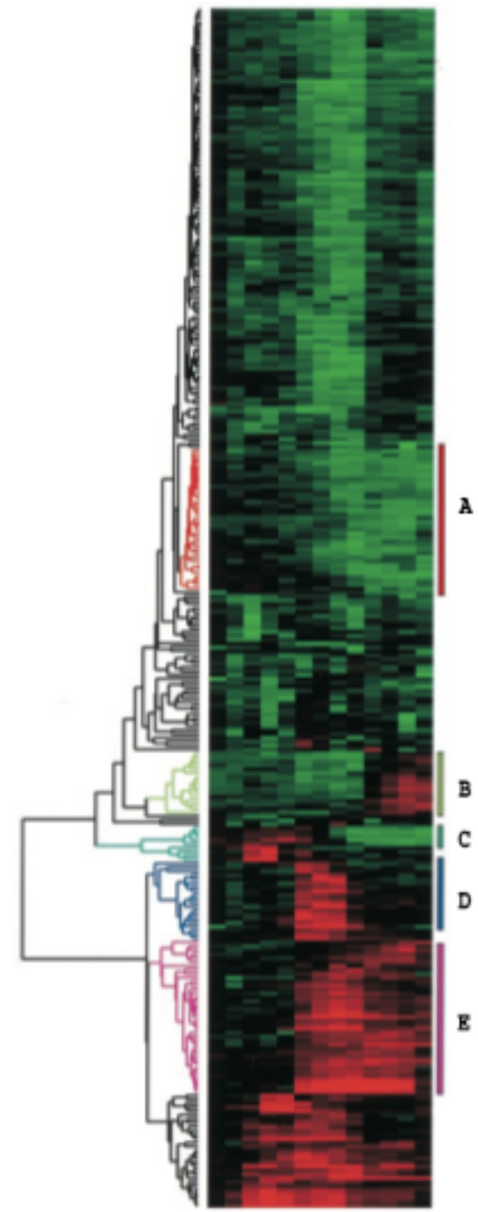
Clustering and genes

Eisen et al., Proc. Natl .Acad. Sci USA, Vol. 95, p. 14863 (1998)



Raw data

Average-link HAC



Clustered and ordered data

Clustering and news

- Suppose you have a bunch (a corpus) of news articles, and you wish to determine which ones are addressing the same topic, rank the topics by 'importance', extract representative articles, etc.
- You wish to do this automatically, i.e. without reading (and understanding) the articles.
- Moreover, you do not know in advance the list of the topics that the articles may address.

If you can define a meaningful distance between the articles, you can then attempt to cluster them, look at the size of the main clusters, select the 'centroids'

This is an application of 'information retrieval' techniques

Clustering and documents

One way of deciding what an article is about, and its relations with others, is to look at the words it contains ('bag of words' model), and check how similar they are to those other articles.

In order to do this, one must assign a **weight** to each word in a document.

A commonly used weight is the so-called **'term frequency-inverted document frequency' (tf-idf)**

[This contrived choice helps reducing the weight of words that occur too often in a collection and that are therefore not relevant for discrimination]

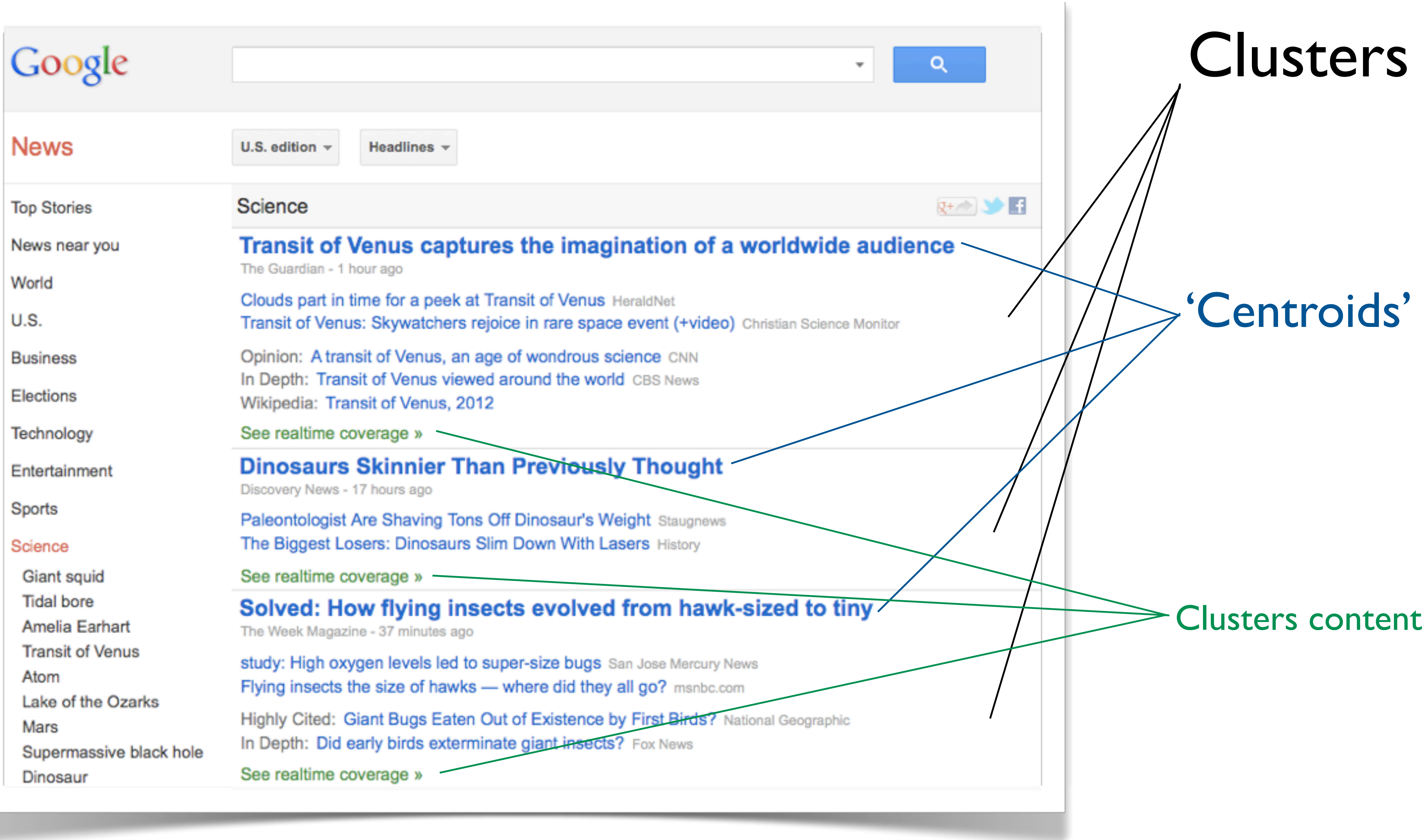
$$\text{tf-idf}_{t,d} = \underbrace{\text{tf}_{t,d}}_{\text{frequency of word t in document d}} \times \log(\underbrace{N}_{\text{number of documents in collection}}/\underbrace{\text{df}_t}_{\text{number of documents containing the word t}})$$

A vector **V**(d) of all tf-idf is constructed for each document.

The **dissimilarity function** is then given by

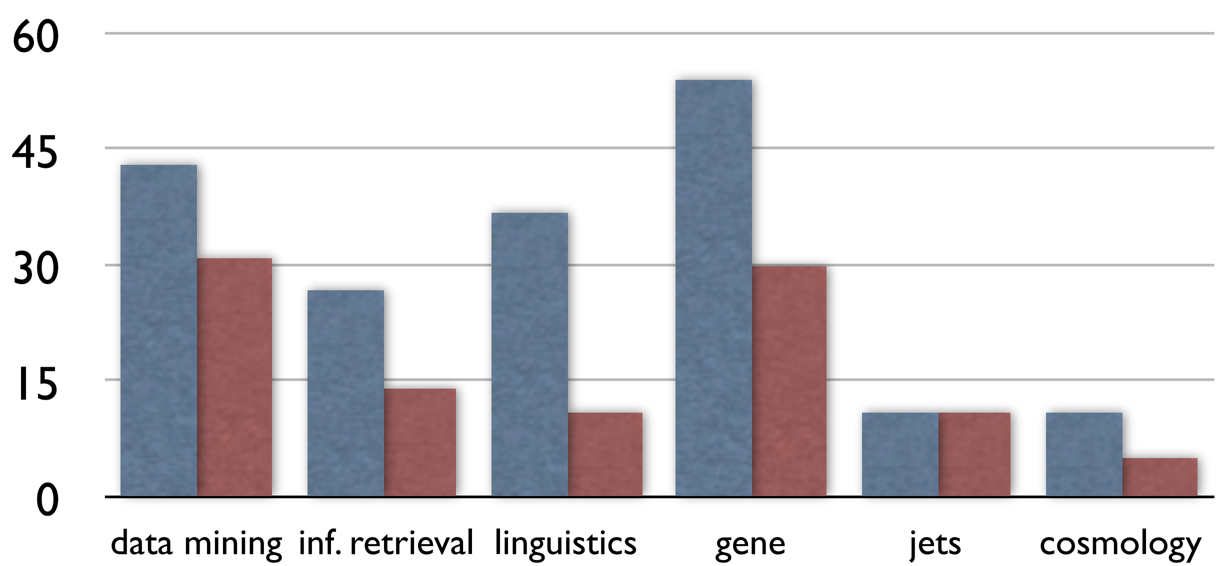
$$D(d_1, d_2) = 1 - \frac{\mathbf{V}(d_1) \cdot \mathbf{V}(d_2)}{|\mathbf{V}(d_1)||\mathbf{V}(d_2)|}$$

Google news



[NB. No guarantee that Google News actually operates precisely this way, but this should give you the correct idea]

Back to "Clustering and..."



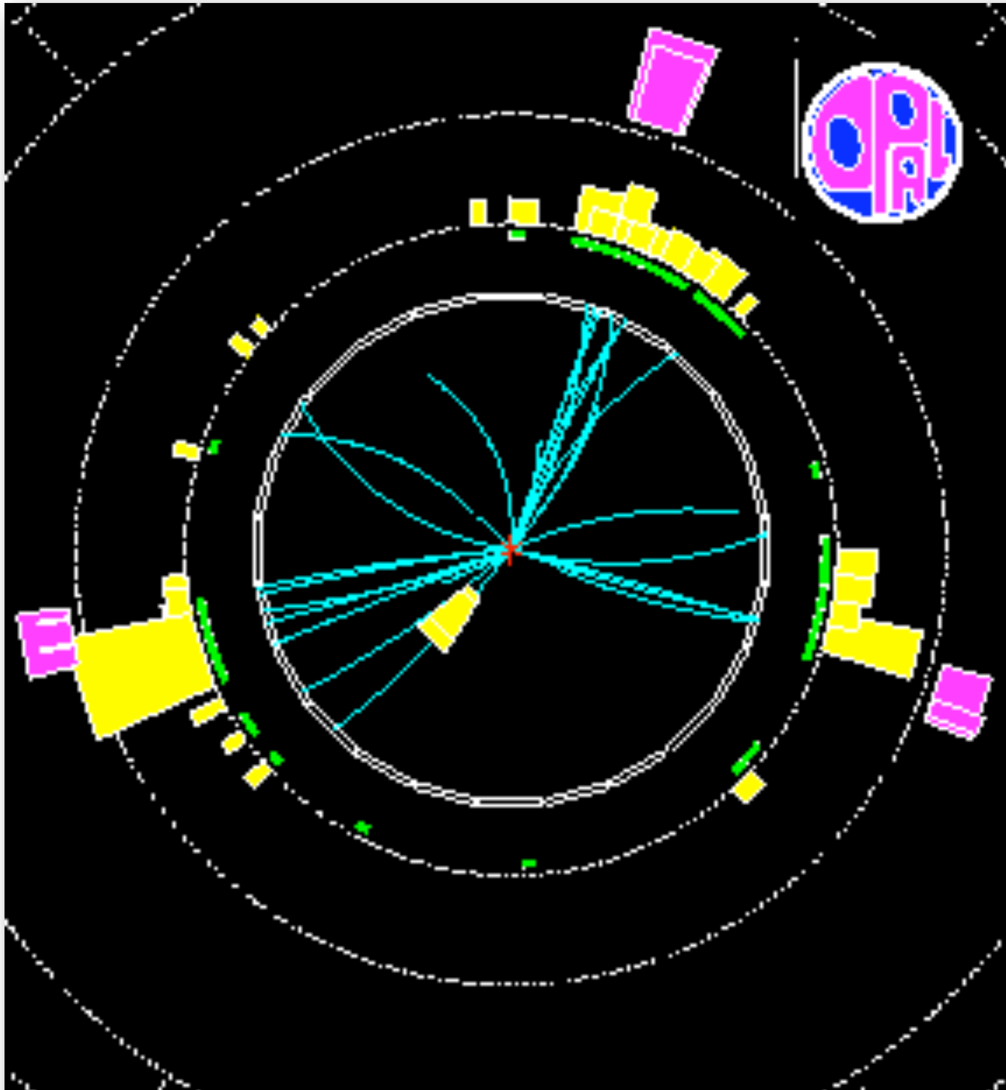
This chart had been produced by selecting 'by hand' some keywords. It is a form of supervised learning, i.e. classification.

Now that we know how to work with documents, we may:

1) Download from Google the 1630k documents containing the word "clustering algorithm"
2) Calculated the td-idf weight for each word in each of them
3) Calculate the dissimilarity matrix using the scalar product between td-idf vectors
4) Run one's favourite algorithm, e.g. HAC with average linkage
5) Decide where to cut in the dendrograms, identify main clusters, look at their size
6) Order the cluster by size (for instance), extract from them the most representative articles

... a lot of work, but done currently (and for free) by search engines

Clusters in HEP: jets



A **jet** is something that happens in high energy events:

a collimated bunch of hadrons flying roughly in the same direction

(though, in the following, we'll extend this intuitive definition somewhat)

Jet clustering algorithm

A **jet algorithm** maps the momenta of the final state particles into the momenta of a certain number of jets:

$$\{p_i\} \xrightarrow{\text{jet algorithm}} \{j_k\}$$

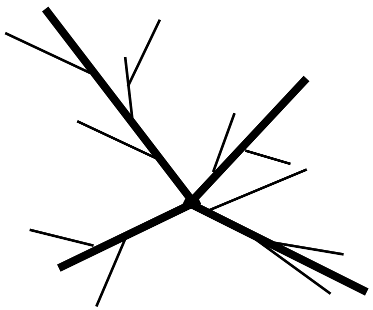
particles,
4-momenta,
calorimeter towers, ...

jets

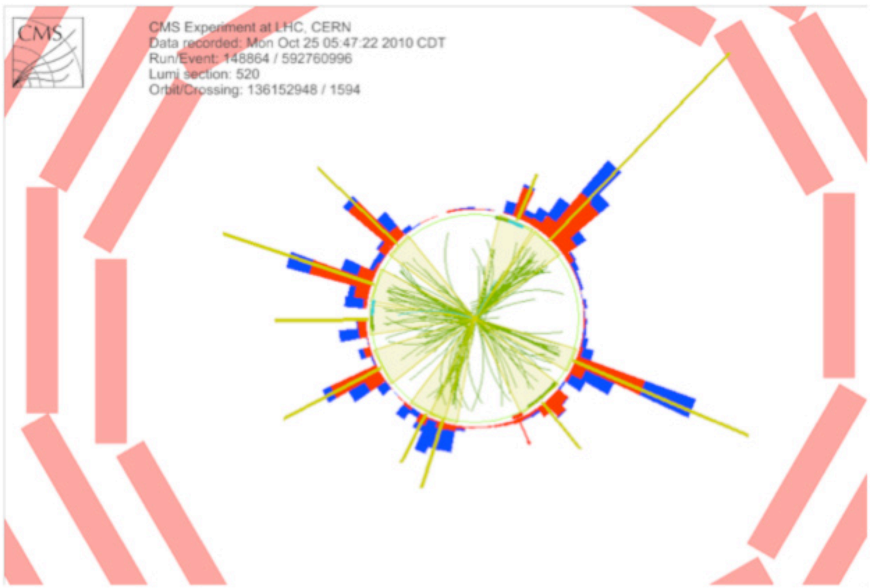
Most algorithms contain a resolution parameter, **R**, which controls the extension of the jet

(more about this later on)

Multileg + PS



??



QCD predictions

Real data

Jets

One purpose of a 'jet clustering' algorithm is to **reduce the complexity** of the final state, simplifying many hadrons to **simpler objects** that one can hope to **calculate**

Jets can serve two purposes

- They can be ***observables***, that one can calculate and measure
- They can be ***tools***, that one can employ to extract specific properties of the final state

What characteristics should they have?

Jets as observables

In physics, a jet is not only a proxy of an initial parton and therefore the result of a generic clustering process

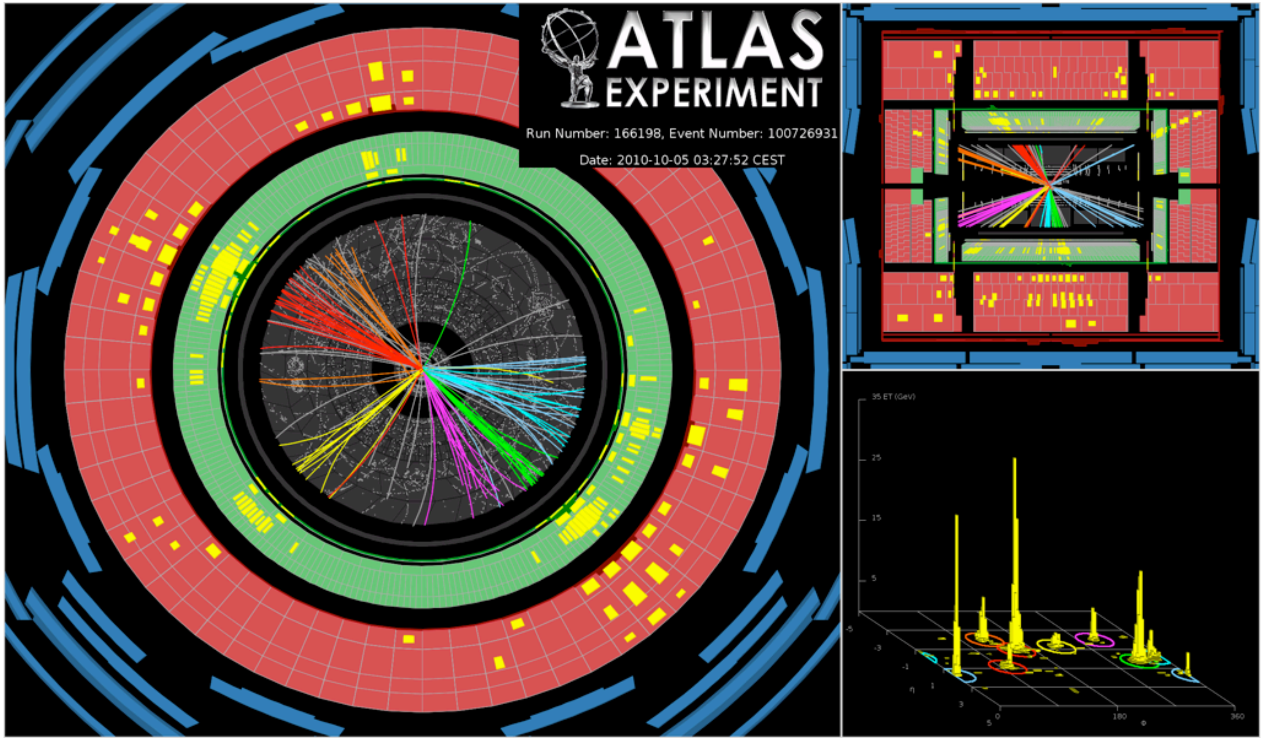
In asking jets to be **useful observables**, we must require that the result be **perturbatively calculable**, so as to exploit the predictive power of QCD

In turn, for an observable to be perturbatively calculable it must be **infrared and collinear (IRC) safe:**
in the limit of a collinear splitting, or the emission of an infinitely soft particle, the observable must remain **unchanged**

Beyond Sterman-Weinberg

Sterman and Weinberg introduced in 1977 an original 'analytical' IRC-safe jet definition, which is useful for 2 (or 3) jets in e^+e^- collisions.

A more complex environment, like hadronic collisions, demands much more flexibility
(e.g., we do not know a priori how many jets to expect)



Hence, the choice to construct jets through a clustering algorithm.

Contrary to other fields, the use made in physics of the clusters (i.e. the jets) puts quite strong requirements on the clustering process

Snowmass

FERMILAB-Conf-90/249-E
[E-741/CDF]

Toward a Standardization of Jet Definitions *

* To be published in the proceedings of the 1990 Summer Study on High Energy Physics, *Research Directions for the Decade*, Snowmass, Colorado, June 25 - July 13, 1990.

Several important properties that should be met by a jet definition are [3]:

1. Simple to implement in an experimental analysis;
2. Simple to implement in the theoretical calculation;
3. Defined at any order of perturbation theory;
4. Yields finite cross section at any order of perturbation theory;
5. Yields a cross section that is relatively insensitive to hadronization.

Speed

Infrared and collinear safety

[Addition of a soft particle or a collinear splitting should not change final hard jets]

Two main classes of jet algorithms

- **Sequential recombination algorithms**

  Bottom-up approach: combine particles starting from **closest ones**

  How? Choose a **distance measure**, iterate recombination until few objects left, call them jets

  Works because of mapping closeness ⇔ QCD divergence

  Examples: Jade, k_t, Cambridge/Aachen, anti-k_t, …..

  → hierarchical clustering

- **Cone algorithms**

  Top-down approach: find coarse regions of energy flow.

  How? Find **stable cones** (i.e. their axis coincides with sum of momenta of particles in it)

  Works because QCD only modifies energy flow on small scales

  Examples: JetClu, MidPoint, ATLAS cone, CMS cone, SISCone…...

  → partitional clustering

Finding stable cones

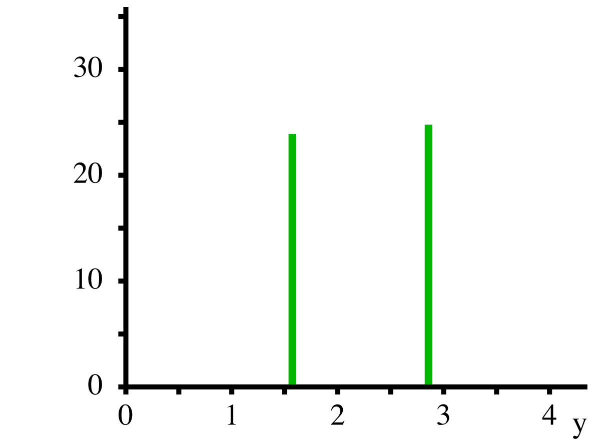
In partitional-type algorithms, one wishes to find
the **stable configurations**:
axis of cones coincides with sum of 4-momenta of particles it contains.

The ‘safe’ way of doing so is to test
all possible combinations of N objects

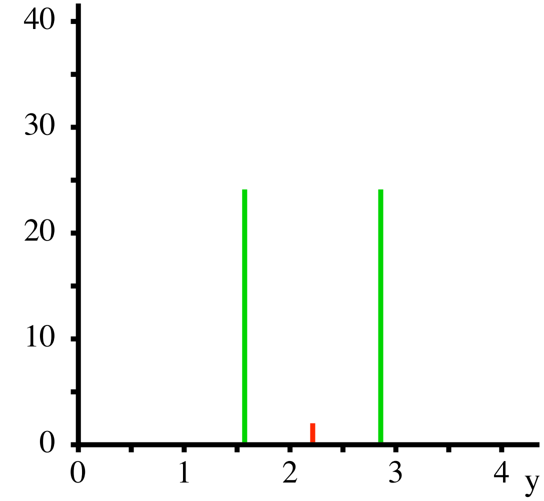
Unfortunately, this takes $N2^N$ operations:
the age of the universe for only 100 objects

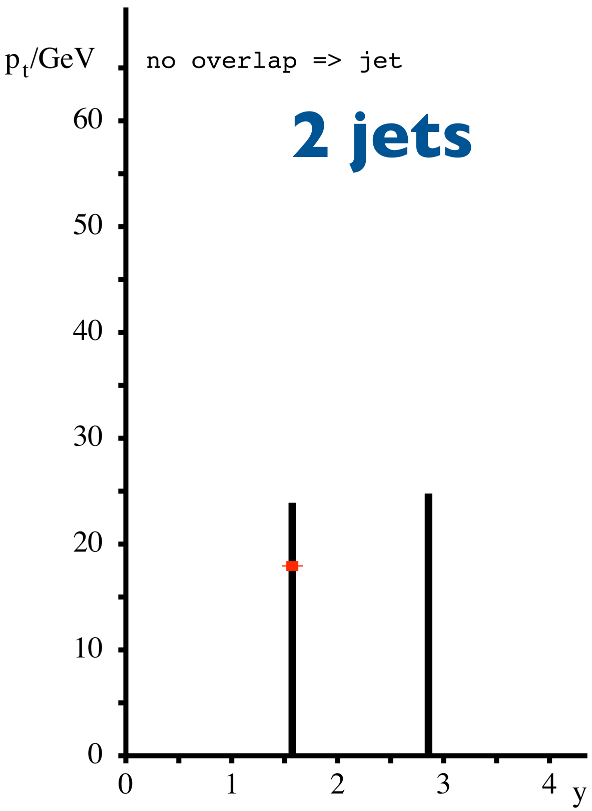
An approximate way out is to use seeds (e.g. à la k-means)
However, the final result can depend on the choice of the seeds and,
such jet algorithms usually turn out to be **IRC unsafe**

Cone infrared unsafety

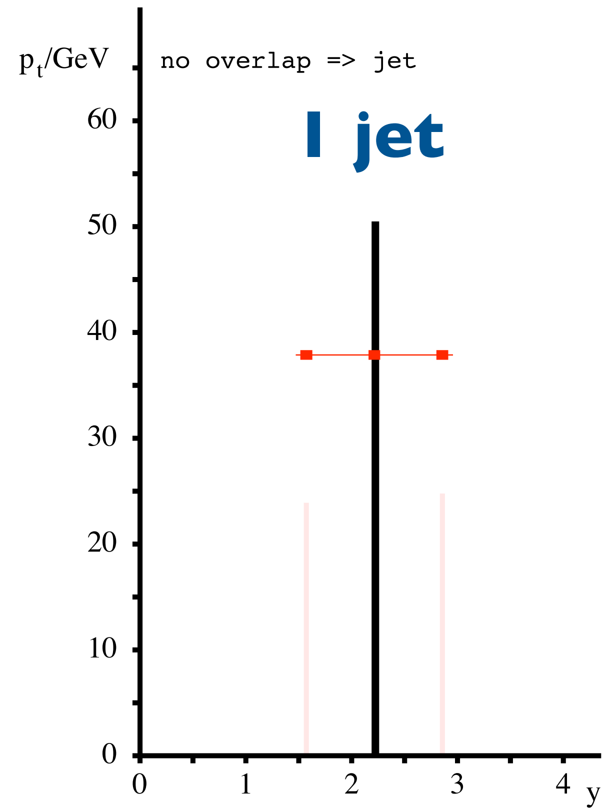


Add a
soft particle





Final state jets differ



Recombination algorithms

- First introduced in e^+e^- collisions in the '80s
- Typically they work by calculating a **'distance'** between particles, and then recombine them pairwise according to a given order, until some condition is met (e.g. no particles are left, or the distance crosses a given threshold)

IRC safety can usually be seen to be trivially guaranteed

JADE algorithm

distance:

$$y_{ij} = \frac{2E_iE_j(1-\cos\theta_{ij})}{Q^2}$$

- Find the minimum y_{min} of all y_{ij}
- If y_{min} is below some jet resolution threshold y_{cut}, recombine i and j into a single new particle ('pseudojet'), and repeat
- If no $y_{min} < y_{cut}$ are left, all remaining particles are jets

Problem of this particular algorithm:
two soft particles emitted at large angle get easily recombined into a single jet: counterintuitive and perturbatively troublesome

e^+e^- k_t (Durham) algorithm

[Catani, Dokshitzer, Olsson, Turnock, Webber '91]

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

In the collinear limit, the numerator reduces to the **relative transverse momentum** (squared) of the two particles, hence the name of the algorithm

The use of the min() avoids the problem of recombination of back-to-back particles present in JADE: a soft and a hard particle close in angle are 'closer' than two soft ones at large angle

One key feature of the k_t algorithm is its relation to the structure of QCD divergences:

$$\frac{dP_{k\to ij}}{dE_i d\theta_{ij}} \sim \frac{\alpha_s}{\min(E_i, E_j)\theta_{ij}}$$

The k_t algorithm inverts the QCD branching sequence (the pair which is recombined first is the one with the largest probability to have branched)

k_t algorithm in hadron collisions

(Inclusive and longitudinally invariant version)

Catani, Dokshitzer, Seymour and Webber, '93
S.Ellis and Soper, '93

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

- Calculate the distances between the particles: $\mathbf{d_{ij}}$
- Calculate the beam distances: $\mathbf{d_{iB}}$
- Combine particles with **smallest distance** d_{ij} or, if d_{iB} is smallest, call it a jet
- Find again smallest distance and repeat procedure until no particles are left (this stopping criterion leads to the *inclusive* version of the k_t algorithm)

The speed 'problem'

Given N particles the k_t algorithm is, naively, an $O(N^3)$ algorithm: calculate N^2 distances, repeat for all N iterations

With 1000 particles (typical LHC event), this takes 10^9 operations, i.e. about a second on a modern GHz CPU

Clustering such an event would take significantly more than generating it in a MonteCarlo, not to speak about trying to use the algorithm at the trigger level, where the time budget is of the order of tens of milliseconds

This, together with the tendency of the k_t algorithm to 'scoop up' soft radiation quite far from the hard partons, and to give jets with ragged borders, difficult to correct for, had led people to prefer cone algorithms in a hadronic environment

The jet revolution

2005: the k_t algorithm is made fast

MC, Salam, hep-ph/0512210

2007: the cone algorithm is made safe (but still fast)

Salam, Soyez, arXiv:0704:0292

In both these cases, key to success was to exploit **geometrical**, rather than combinatorial, methods

2008: jet areas are introduced

MC, Salam, Soyez, arXiv:0802.1188

2008: anti-k_t is invented

MC, Salam, Soyez, arXiv:0802.1189

2008: jet substructure is revived

Butterworh, Davison, Rubin, Salam, arXiv:0802.2470

The FastJet algorithm

Sequential recombination algorithms are computationally heavy because one naively calculates **all distances between all particles** ($O(N^2)$ step), before recombining them ($O(N)$ step)

Considering the problem from a **geometrical** point of view, one realizes that, in the k_t algorithm, when a particle gets combined with another, and has the smallest k_t, **its partner is its geometrical nearest neighbour on the cylinder spanned by y and ϕ**

This means that we need to look for partners only among the near neighbours of all particles:
a few neighbours each × N particles = O(N) operations

The FastJet algorithm

Exploiting the geometrical observation, one can formulate the following implementation of the k_t algorithm:

- For each particle i establish its **geometrical nearest neighbour $\mathbf{G_i}$** and calculate the arrays $\mathbf{d_{iG_i}}$ and $\mathbf{d_{iB}}$
- Find the minimal value $\mathbf{d_{min}}$ of the $\mathbf{d_{iG_i}}$ and $\mathbf{d_{iB}}$, combine particles corresponding to it
- Update $\mathbf{d_{iG_i}}$ and $\mathbf{d_{iB}}$ if needed, continue until no particles left

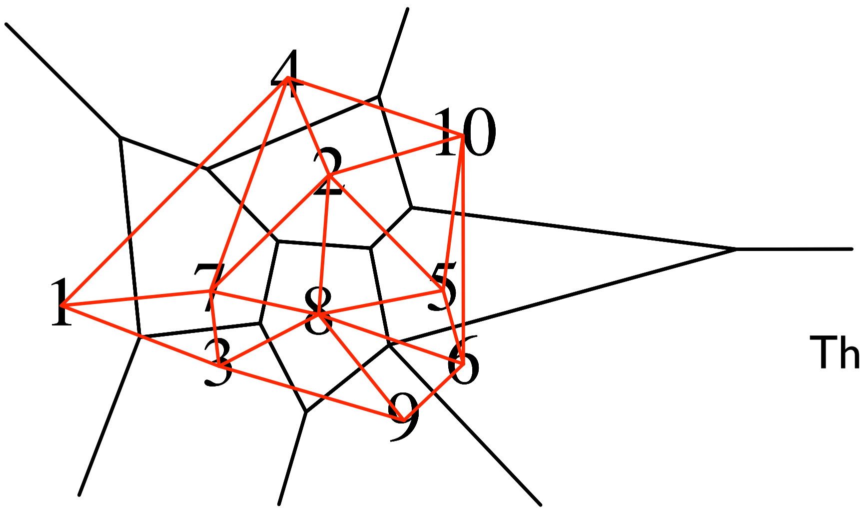
This is already an **$\mathbf{O(N^2)}$ algorithm**: find a few neighbours for each of the N particles, and repeat N times to recombine them all.

But one can do better

FJ: the Voronoi implementation

Our problem has now become a **geometrical** one:
how to find efficiently the (nearest) neighbour(s) of a point

Widely studied problem in computational geometry. Tool: **Voronoi diagram**



Definition: each cell contains the locations which have the given point as nearest neighbour.

The **dual** of a Voronoi diagram is a **Delaunay triangulation**

Once the Voronoi diagram is constructed, **the nearest neighbour of a point will be in one of the O(1) cells sharing an edge with its own cell**

Example: the G(eometrical) N(earest) N(eighbour) of point 7 will be found among 1,4,2,8 and 3 (it turns out to be 3)

FJ: the Voronoi implementation

MC and G.P. Salam, hep-ph/0512210

Construct the Voronoi diagram of the N particles
(i.e. using the CGAL library) **O(N lnN)**

Find the GNN of each of the N particles. Construct the d_{ij} distances, store the results in a priority queue (i.e. a C++ map) **O(N lnN)**

Merge/eliminate particles appropriately

Update Voronoi diagram and distances' map **O(lnN)**

repeat N times

Overall, an O(N ln N) algorithm

The k_t algorithm and its siblings

$$d_{ij} = \min(k_{ti}^{2p}, k_{tj}^{2p}) \frac{\Delta y^2 + \Delta\phi^2}{R^2} \qquad d_{iB} = k_{ti}^{2p}$$

p = 1 k_t algorithm

S. Catani, Y. Dokshitzer, M. Seymour and B. Webber, Nucl. Phys. B406 (1993) 187
S.D. Ellis and D.E. Soper, Phys. Rev. D48 (1993) 3160

p = 0 Cambridge/Aachen algorithm

Y. Dokshitzer, G. Leder, S.Moretti and B. Webber, JHEP 08 (1997) 001
M.Wobisch and T.Wengler, hep-ph/9907280

p = -1 **anti-k_t algorithm**

MC, G. Salam and G. Soyez, arXiv:0802.1189

NB: in anti-kt pairs with a **hard** particle will cluster first: if no other hard particles are close by, the algorithm will give **perfect cones**

Quite ironically, a sequential recombination algorithm is the 'perfect' cone algorithm

IRC safe algorithms

k_t	SR $d_{ij} = \min(k_{ti}^2,k_{tj}^2)\Delta R_{ij}^2/R^2$ hierarchical in rel p_t	Catani et al '91 Ellis, Soper '93	NlnN
Cambridge/Aachen	SR $d_{ij} = \Delta R_{ij}^2/R^2$ hierarchical in angle	Dokshitzer et al '97 Wengler, Wobish '98	NlnN
anti-k_t	SR $d_{ij} = \min(k_{ti}^{-2},k_{tj}^{-2})\Delta R_{ij}^2/R^2$ gives perfectly conical hard jets	MC, Salam, Soyez '08 (Delsart, Loch)	$N^{3/2}$
SISCone	Seedless iterative cone with split-merge gives 'economical' jets	Salam, Soyez '07	$N^2\ln N$

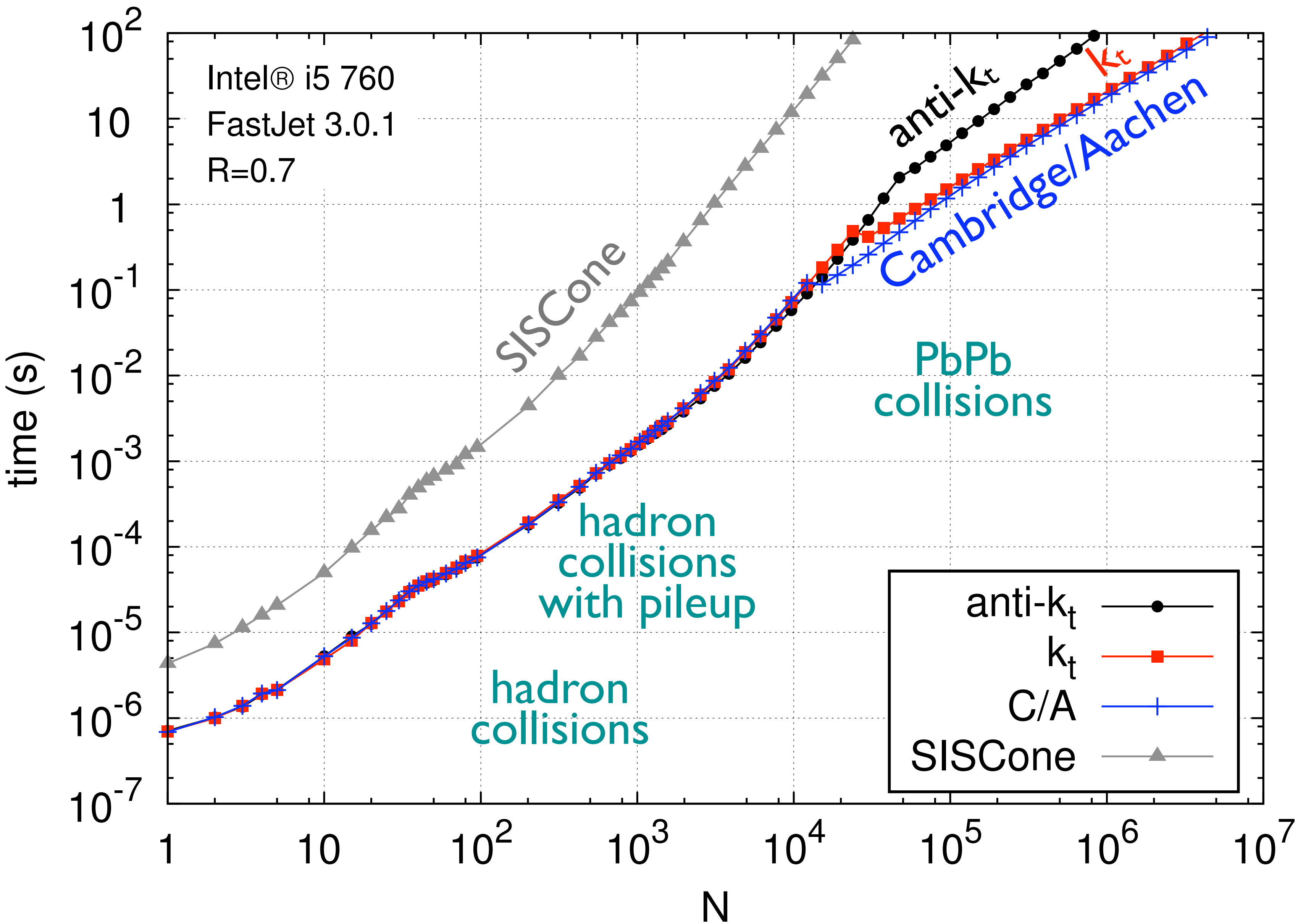
'second-generation' algorithms

All are available in FastJet, http://fastjet.fr

(As well as many IRC unsafe ones)

FastJet speed test

Time needed to cluster an event with N particles



Jets ‘reach’

Algorithmically, a jet is simply a collection of particles. It is however useful to consider its **spatial extent**, i.e. given the position of its axis, **up to where does it collect particles**? What is its **shape**?

These details are important for understanding (and possibly correcting for) number of physical effects of various origin: perturbative, non-perturbative (hadronisation), detector related, etc

In order to describe quantitatively these features, one can **define** the concept of **jet area**

From jet 'reach' to jet areas

Not one, but three **definitions** of a jet's size:

MC, Salam, Soyez, arXiv:0802.1188

- **Passive** area

  Place a single 'ghost' particle in the event, measure the extent of the region where it gets clustered within a given jet

  Reach of jet for **pointlike** radiation

- **Active** area

  Fill the events with many 'ghost' particles, cluster them together with the real ones, see how many get clustered within a given jet

  Reach of jet for **diffuse** radiation

- **Voronoi** area

  Sum of areas of intersections of Voronoi cells of jet constituents with circle of radius R centred on each constituent

  Coincides with passive area for k_t algorithm

(In the large number of particles limit all areas converge to the same value)

Jet active area

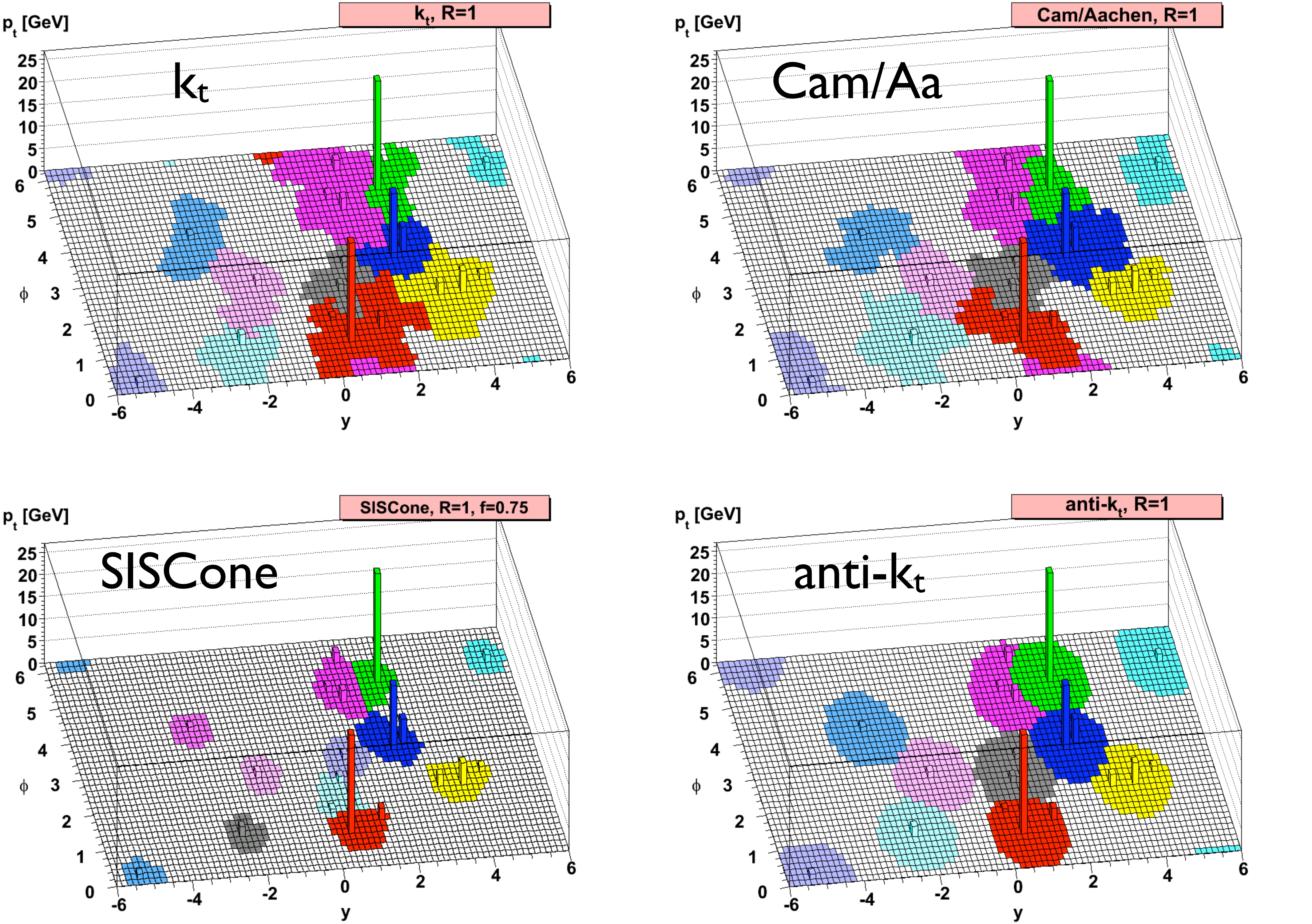
The definition of **active area** mimics the behaviour of the jet-clustering algorithms in the presence of a **large number of randomly distributed soft particles,** like those due to **pileup or underlying event**

Tools needed to implement it

1. An **infrared safe jet algorithm** (the ghosts should not change the jets)
2. A reasonably **fast implementation** (we are adding thousands of ghosts)

Both are available

As a bonus, active areas also allow for a **visualisation** of a jet's reach



kt, R=1
kt
Cam/Aachen, R=1
Cam/Aa
SISCone, R=1, f=0.75
SISCone
anti-kt, R=1
anti-kt
pt [GeV]
0
5
10
15
20
25
φ
y
-6
-4
-2
0
2
4
6
1
3

Jet areas: the single hard particle case

A jet of 'radius' R will surely have area πR^2, right?

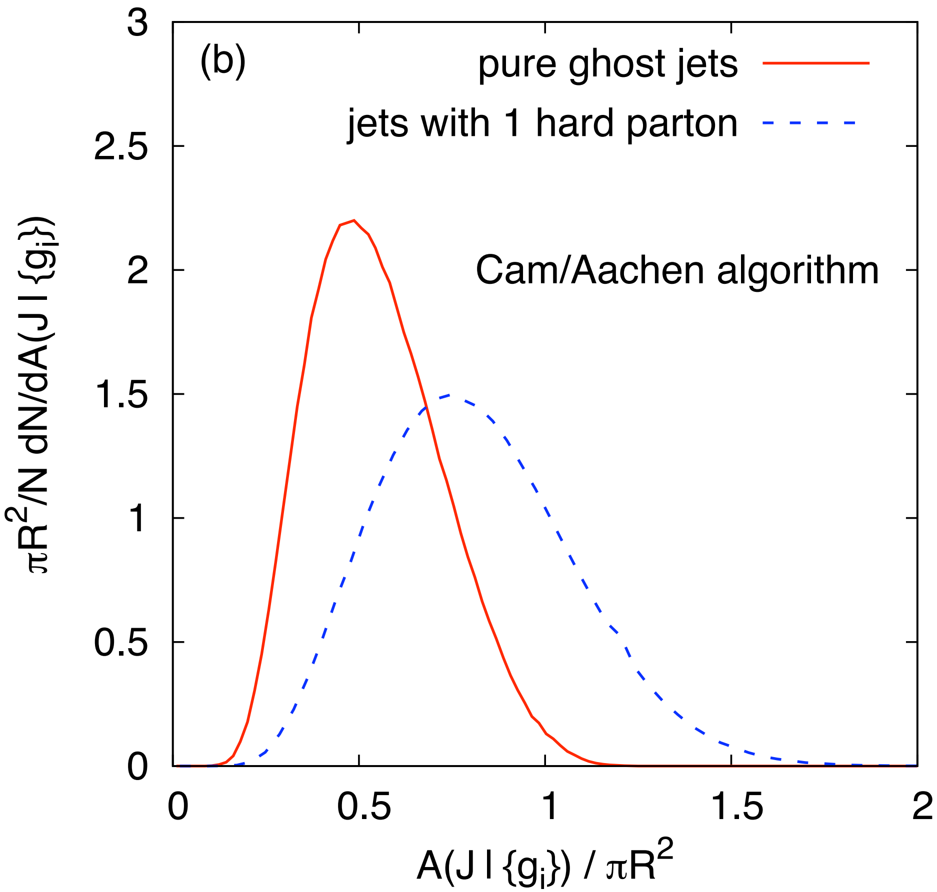
For a jet made of a single hard particle,
passive areas are indeed $\mathbf{\pi R^2}$, but **active** areas are not

Active areas	k_t	Cam/Aa	SISCone	anti-k_t
$\langle A\rangle/\pi R^2$	0.81	0.81	1/4	1

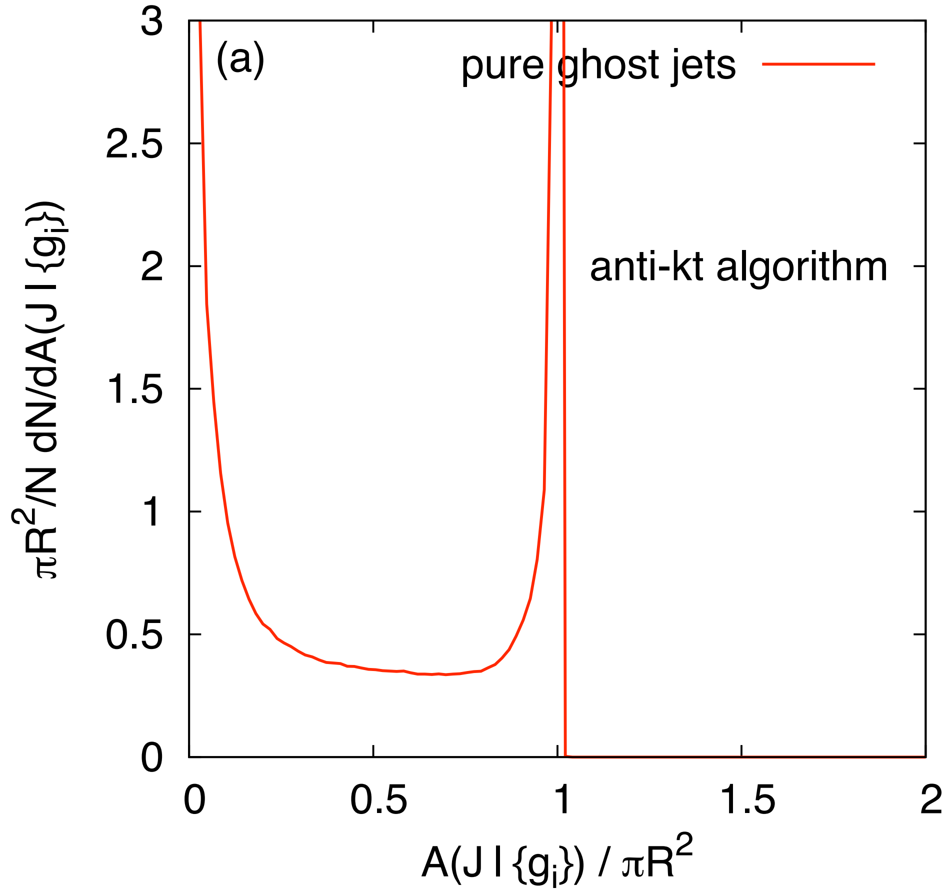
Only anti-k_t has the behaviour one would naively expect

Active area distributions

k_t and Cam/AA



anti-k_t



For a roughly uniformly soft background, anti-k_t gives many small jets and many large ones

(you can't fill a plane with circles!)

A challenge

Can we predict analytically the active area distribution or, at least, its average, for the self-clustering of a large number of 'particles'?

Consider a simplified toy-model: 1-dimensional clustering, and an "euclidean" recombination scheme: the recombination of two 'particles' is simply the mid-point of their coordinate.

N 'particles' are distributed randomly over a length L, such that

$$a \equiv L/N \ll 2R \ll L$$

where R is the 'radius' parameter of the Cambridge-like clustering algorithm

Upon clustering, we get n_J jets, with number of constituents n_C.
The average 'area' (over many events) of a jet will be $\langle A \rangle = a \langle n_C \rangle$

Can we calculate analytically $\displaystyle \lim_{a/2R \to 0, 2R/L \to 0} \frac{\langle A \rangle}{2R}$?

(In this toy model, or in a similarly simple one. What about in higher dimensions?)

Areas as a dynamical jet property

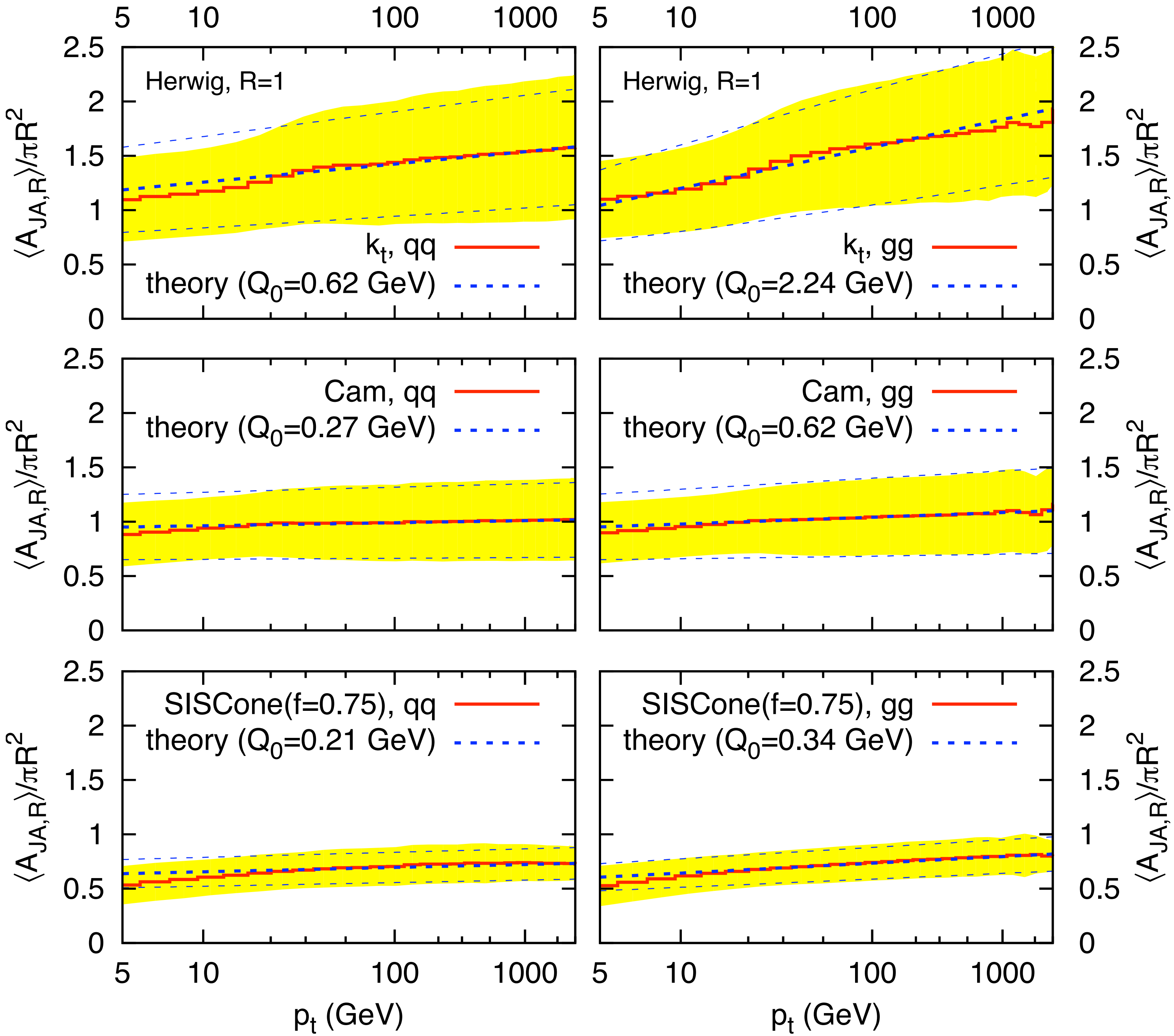
The area of a jet can change with its p_t:

$$\langle \Delta A \rangle = \mathrm{D} \, \frac{C_1}{\pi b_0} \ln \frac{\alpha_s(Q_0)}{\alpha_s(R p_{t1})}$$

	k_t	Cam/Aa	SISCone	anti-k_t
D	0.52	0.08	0.12	0

Again, only anti-k_t has a typical area that does **not** increase with p_t

Jet areas scaling violations



Averages and dispersions evolution from Monte Carlo simulations (dijet events at LHC) in good agreement with simple LL calculations

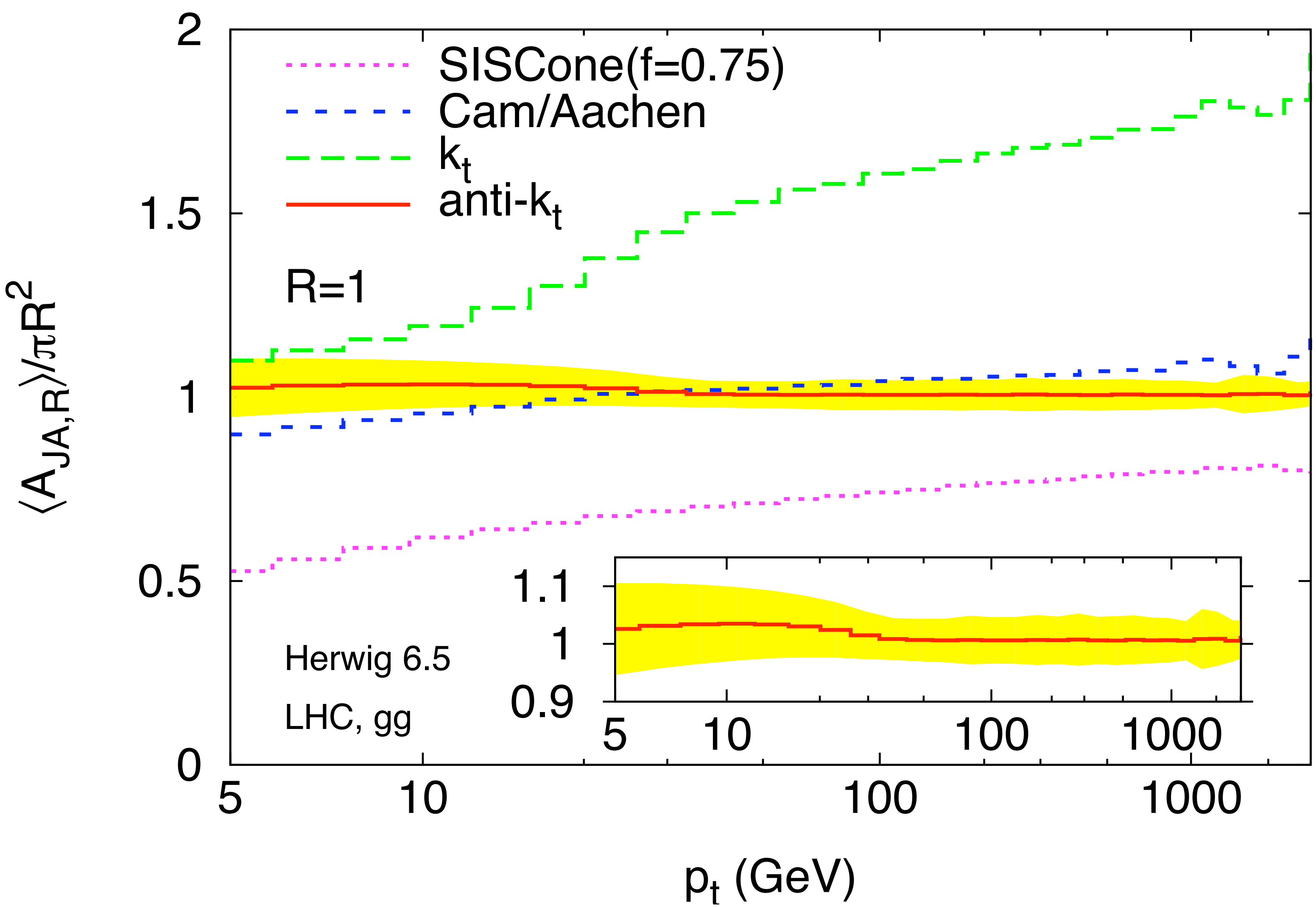
Area scaling violations are a legitimate observable

(Though they may not be the best place where to measure α_s ....)

Jet areas scaling violations

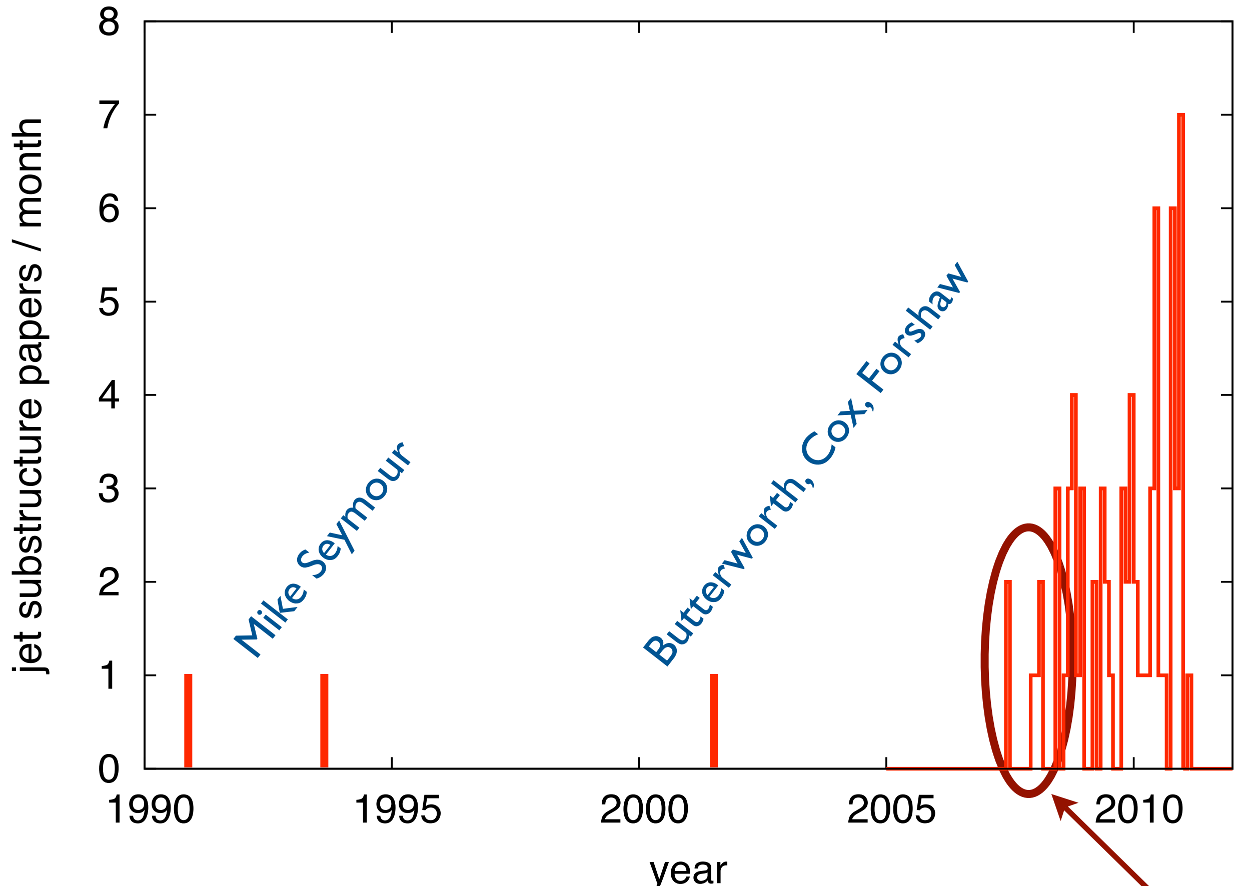
MC, Salam, Soyez, arXiv:0802.1189

Check anti-k_t behaviour: scaling violations indeed absent, as predicted



The frontier: jet substructure

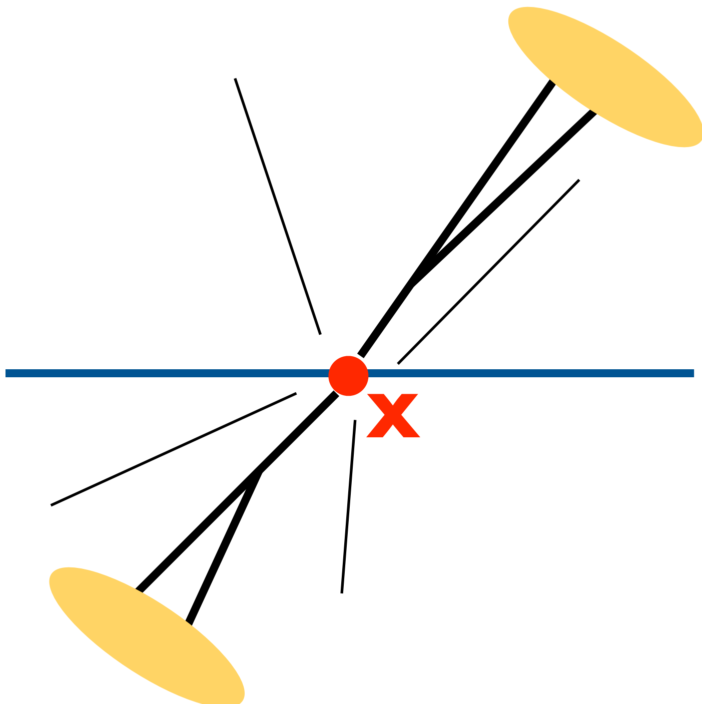
Number of papers containing the words 'jet substructure' and 'LHC' in INSPIRE



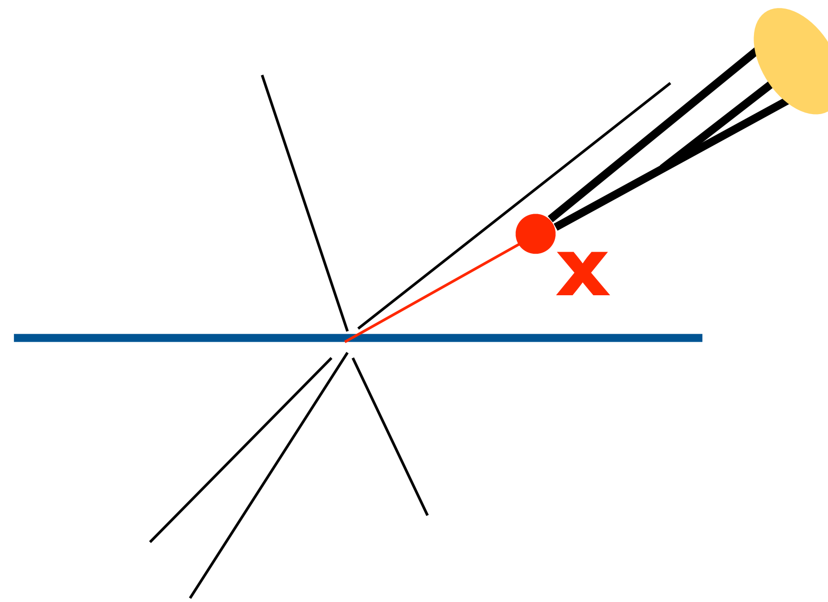
More than 70 papers after 2008.
Essentially none before

15. **Jet substructure as a new Higgs search channel at the LHC.**
Jonathan M. Butterworth, Adam R. Davison (University Coll. London), Mathieu Rubin, Gavin P. Salam (Paris, LPTHE).
Published in **Phys.Rev.Lett. 100 (2008) 242001**
e-Print: **arXiv:0802.2470 [hep-ph]**

Why boosted objects



Heavy particle X at **rest**



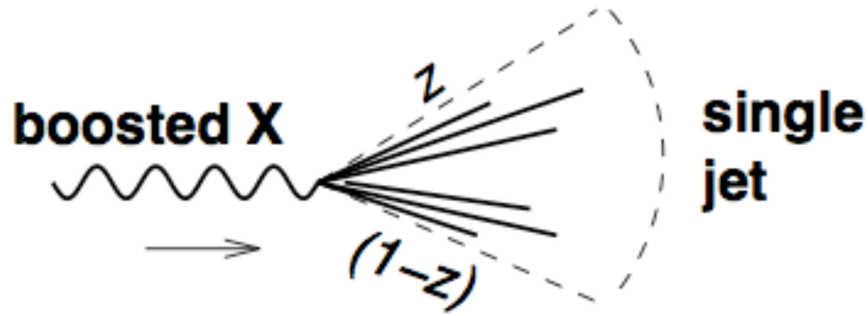
Boosted heavy particle X

Easy to resolve jets and calculate invariant mass, but signal very likely swamped by background (eg H→bb v. tt →WbWb)

Cross section very much reduced, but acceptance better and some backgrounds smaller/ reducible

Why substructure

Scales: m ~ 100 GeV, p_t ~ 500 GeV



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

- need small R (< $2m/p_t$ ~ 0.4) to resolve two prongs
- need large R (>~ $3m/p_t$ ~ 0.6) to cluster into a single jet

Possible strategies

- Use large R, get a single jet : background large
- Use small R, resolve the jets : what is the right scale?
- Let an algorithm find the 'right' substructure

The strategy

A generic substructure approach will

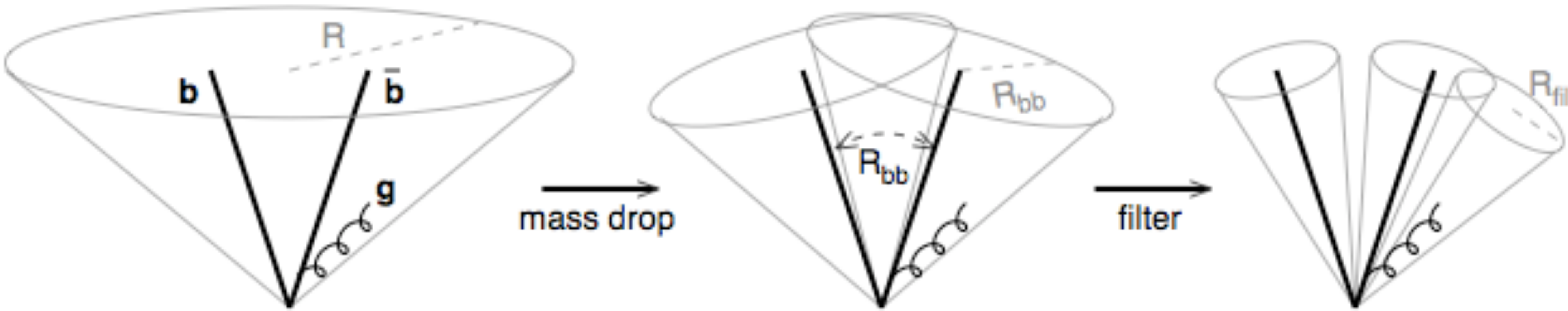
- Cluster initially with a large R, so as to collect all the decay products of a boosted heavy particle into a single jet
- Decluster this jet into subjets, using some condition to decide when to stop the declustering (i.e. find the 'relevant splitting'), possibly including kinematical cuts to reduce the QCD background.

  The stopping condition automatically finds the 'right size' for the distance between the two prongs of the heavy particle decay

The BDRS tagger

These ideas led to the first 'modern' implementation of a boosted tagger

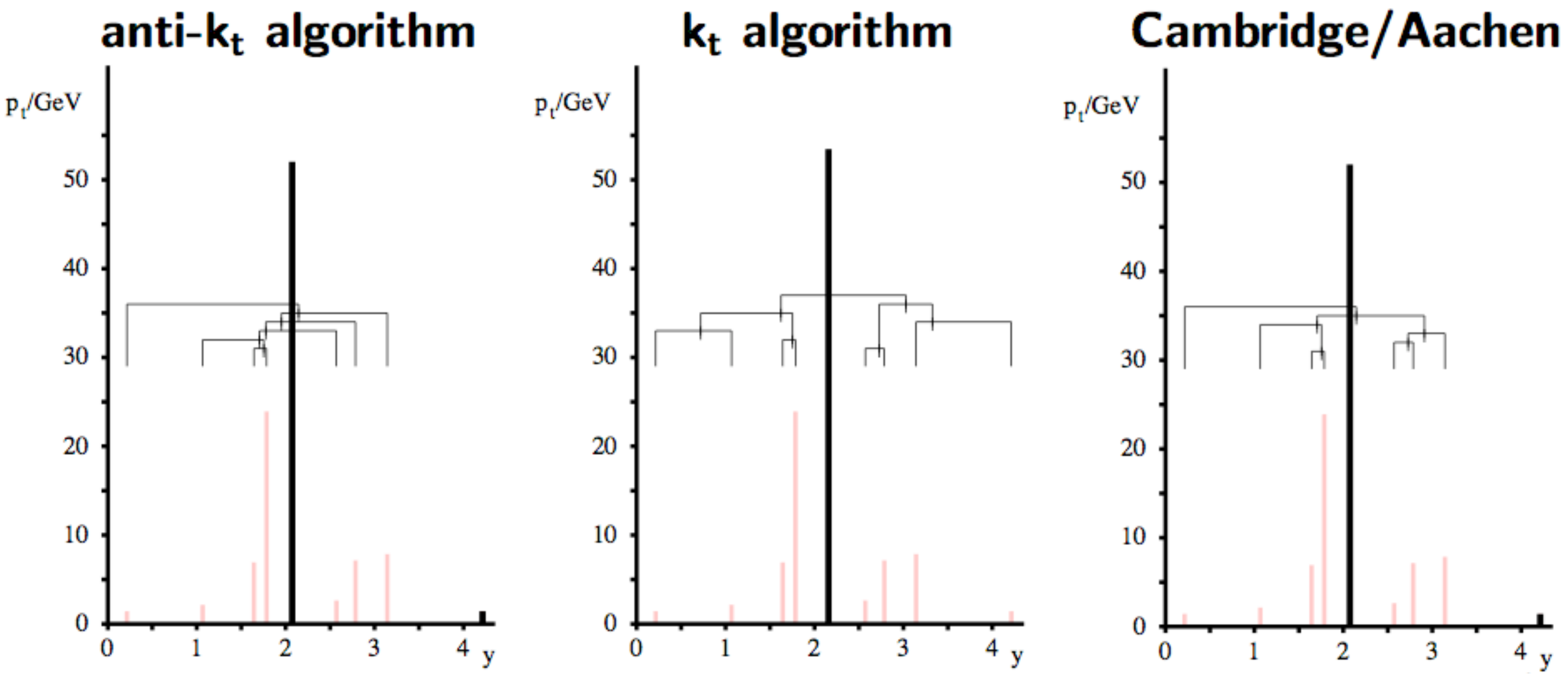
15. **Jet substructure as a new Higgs search channel at the LHC.**
Jonathan M. Butterworth, Adam R. Davison (University Coll. London), Mathieu Rubin, Gavin P. Salam (Paris, LPTHE).
Published in **Phys.Rev.Lett. 100 (2008) 242001**
e-Print: **arXiv:0802.2470 [hep-ph]**

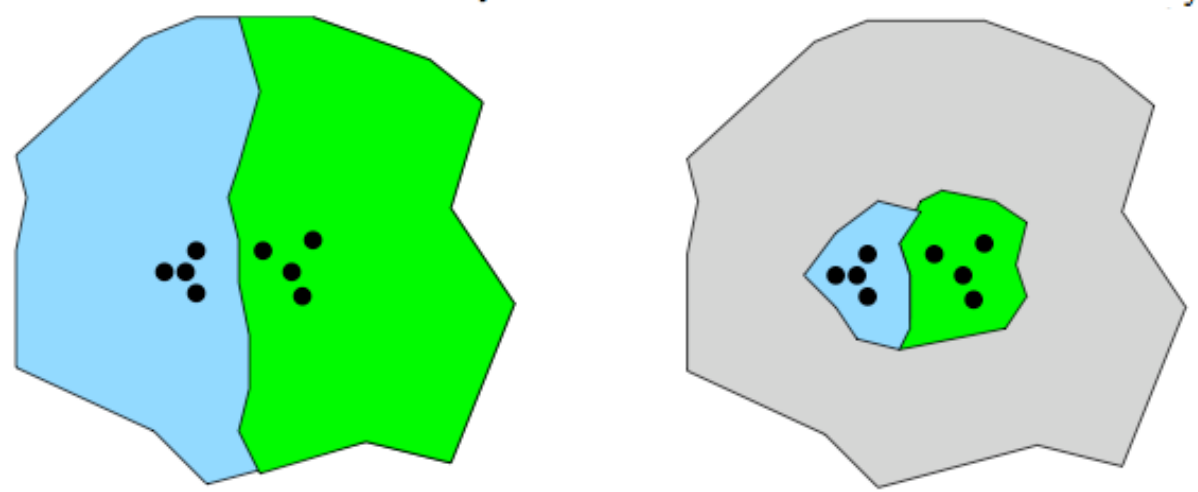


It is a two-prongs tagger for boosted Higgs, which

- Uses the Cambridge/Aachen algorithm (see why in the next slide)
- Employs a Mass-Drop condition (as well as an asymmetry cut) to find the relevant splitting (i.e. 'tag' the heavy particle)
- Includes a post-processing step, using '**filtering**' (introduced in the same paper) to clean as much as possible the resulting jets of UE contamination

Hierarchical substructure





Slide by
Gavin Salam

Phys.Rev.Lett. 100 (2008) 242001

The BDRS tagger

Jet substructure as a new Higgs search channel at the LHC

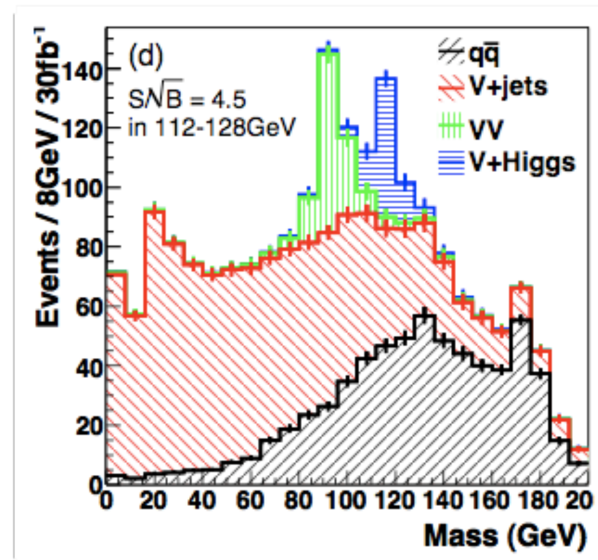
Jonathan M. Butterworth, Adam R. Davison
Department of Physics & Astronomy, University College London.

Mathieu Rubin, Gavin P. Salam
LPTHE; UPMC Univ. Paris 6; Univ. Denis Diderot; CNRS UMR 7589; Paris, France.

It is widely considered that, for Higgs boson searches at the Large Hadron Collider, WH and ZH production where the Higgs boson decays to $b\bar{b}$ are poor search channels due to large backgrounds. We show that at high transverse momenta, employing state-of-the-art jet reconstruction and decomposition techniques, these processes can be recovered as promising search channels for the standard model Higgs boson around 120 GeV in mass.

Jet definition	σ_S/fb	σ_B/fb	$S/\sqrt{B \cdot \text{fb}}$
C/A, $R = 1.2$, MD-F	0.57	0.51	0.80
$K_\perp$, $R = 1.0$, y_{cut}	0.19	0.74	0.22
SISCone, $R = 0.8$	0.49	1.33	0.42

TABLE I: Cross section for signal and the Z+jets background in the leptonic Z channel for $200 < p_{TZ}/\text{GeV} < 600$ and $110 < m_J/\text{GeV} < 125$, with perfect b-tagging; shown for our jet definition, and other standard ones at near optimal R values.



Concluding remarks

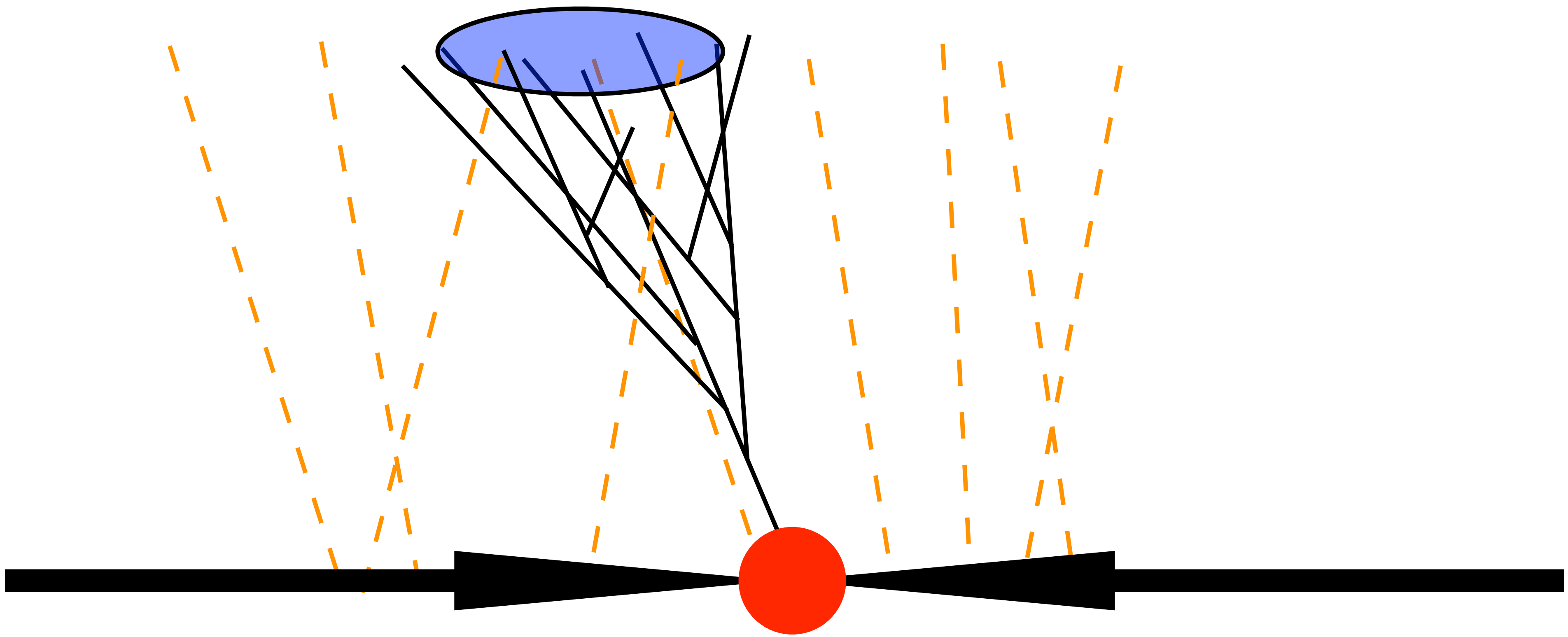
After undergoing a quick phase transition a few years ago, jet clustering seems to have reached stability. All LHC collaborations use anti-k_t, even if with different radii
(eg. CMS 0.5 and 0.7, ATLAS 0.4 and 0.6!)

It is now jet substructure which is an active research topic in rapid development. Many new physics search strategies based on it are now being developed and tested at the LHC.

Given how much the use of clustering is widespread in many disciplines, one can wonder if further new ideas can be either imported or, perhaps, exported.

Backup slides

Hard jets and background



In a realistic set-up underlying event (UE) and pile-up (PU) from multiple collisions produce many soft particles which can ‘contaminate’ the hard jet

Hard jets and background

How are the hard jets modified by the background?

Susceptibility
(how much bkgd gets picked up)

Jet areas

Resiliency
(how much the original jet changes)

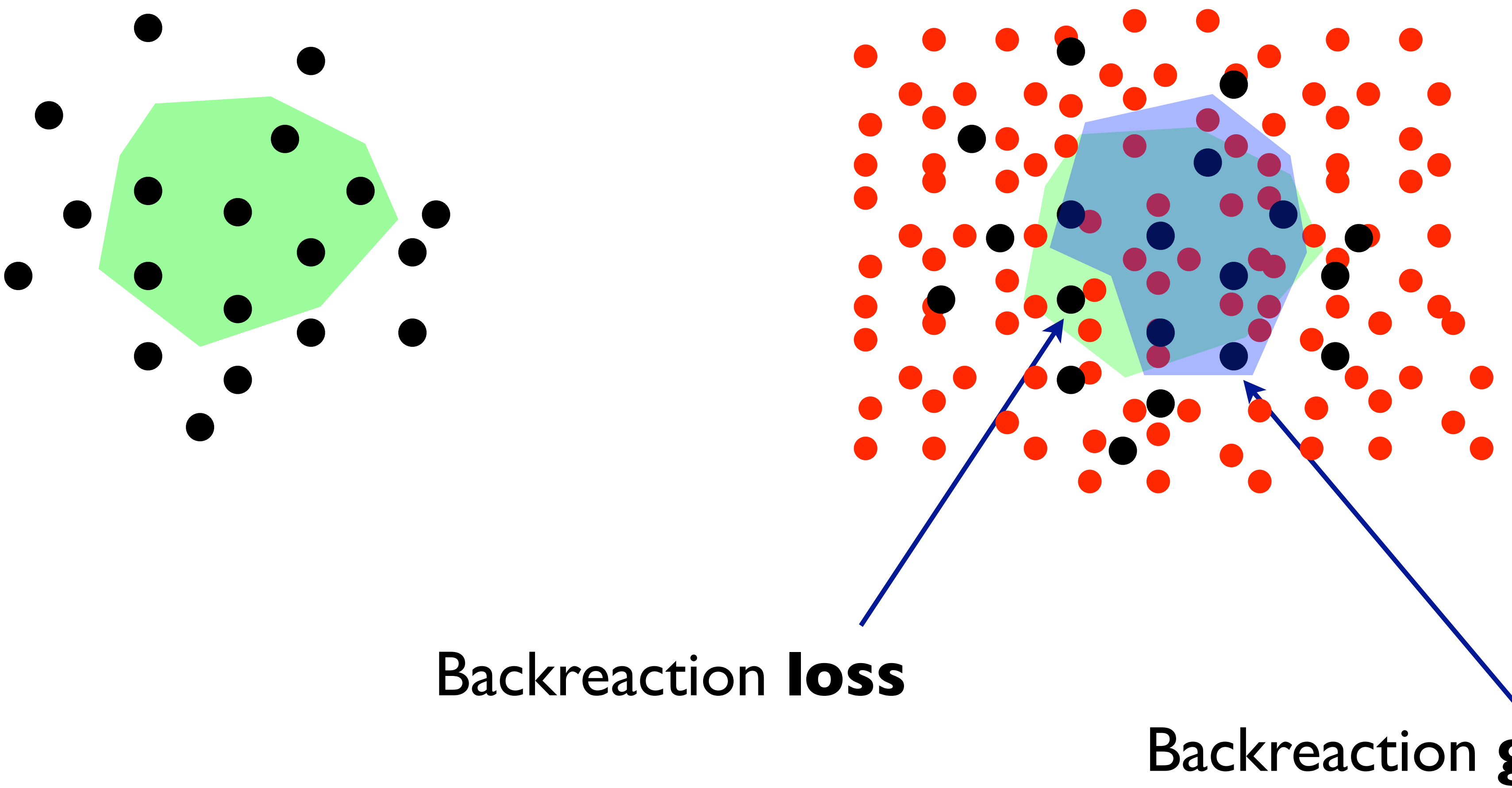
Backreaction

Resiliency: backreaction

"How (much) a jet changes when immersed in a background"

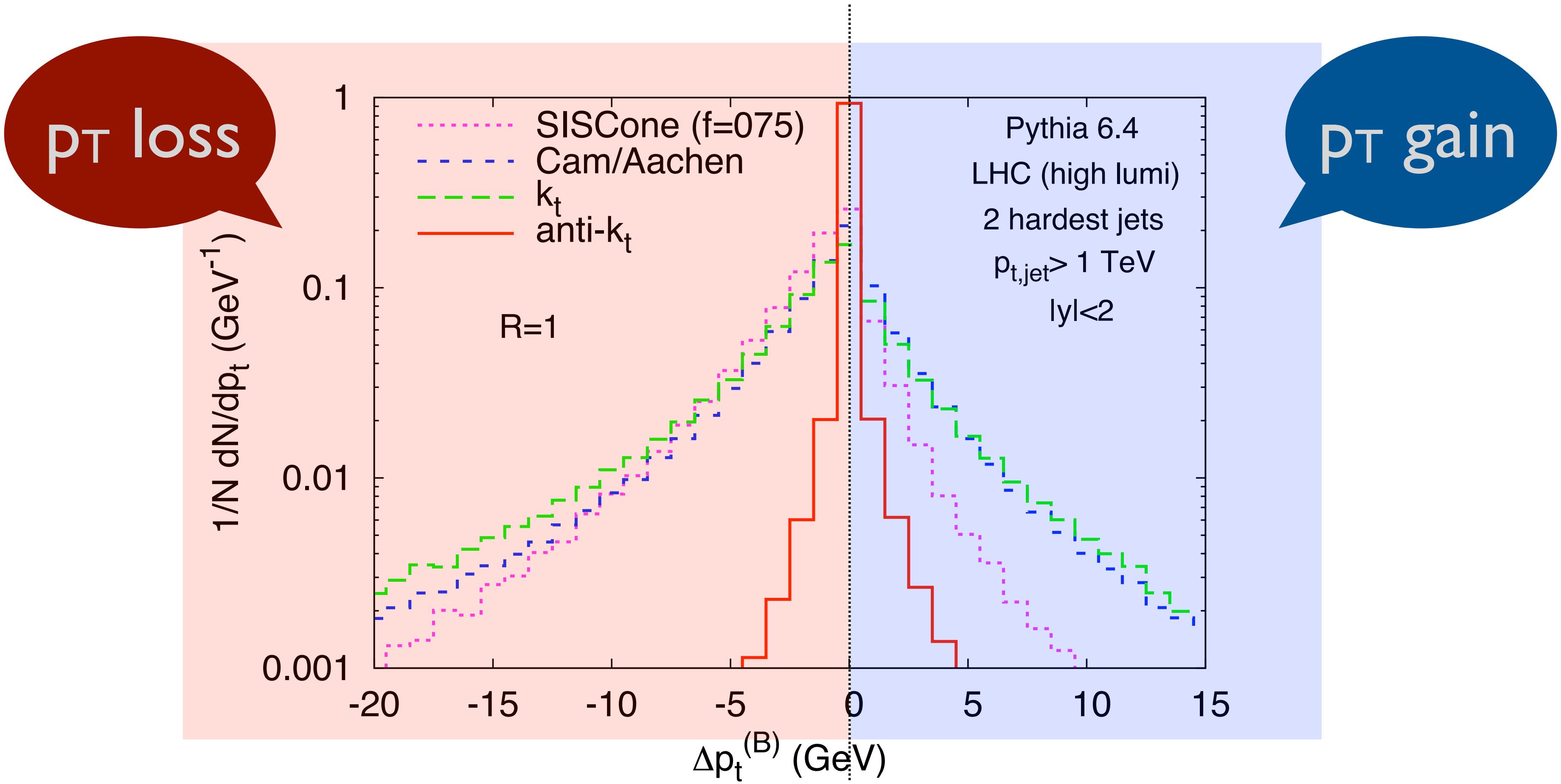
Without background

With background



Resiliency: backreaction

MC, Salam, Soyez, arXiv:0802.1188



Anti-k_t jets are much more resilient to changes from background immersion

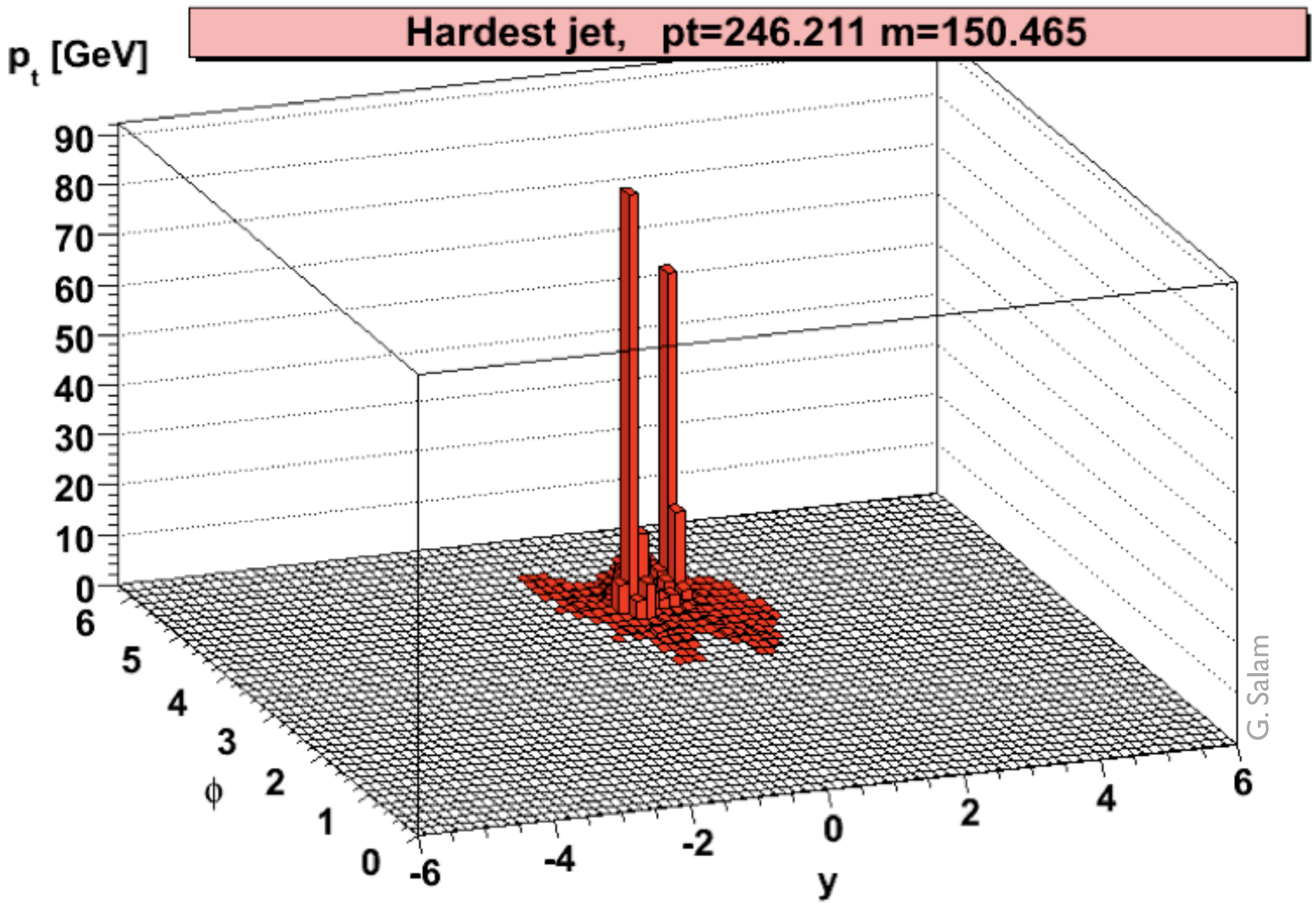
The IRC safe algorithms

	Speed	Regularity	UE contamination	Backreaction	Hierarchical substructure
k_t	☺ ☺ ☺	☂	☂☂	☁☁	☺ ☺
Cambridge /Aachen	☺ ☺ ☺	☂	☂	☁☁	☺ ☺ ☺
anti-k_t	☺ ☺ ☺	☺ ☺	☁/ ☺	☺ ☺	✗
SISCone	☺	☁	☺ ☺	☁	✗

Boosted Higgs tagger

$pp \to ZH \to \nu\bar{\nu}b\bar{b}$

Butterworth, Davison, Rubin, Salam, 2008



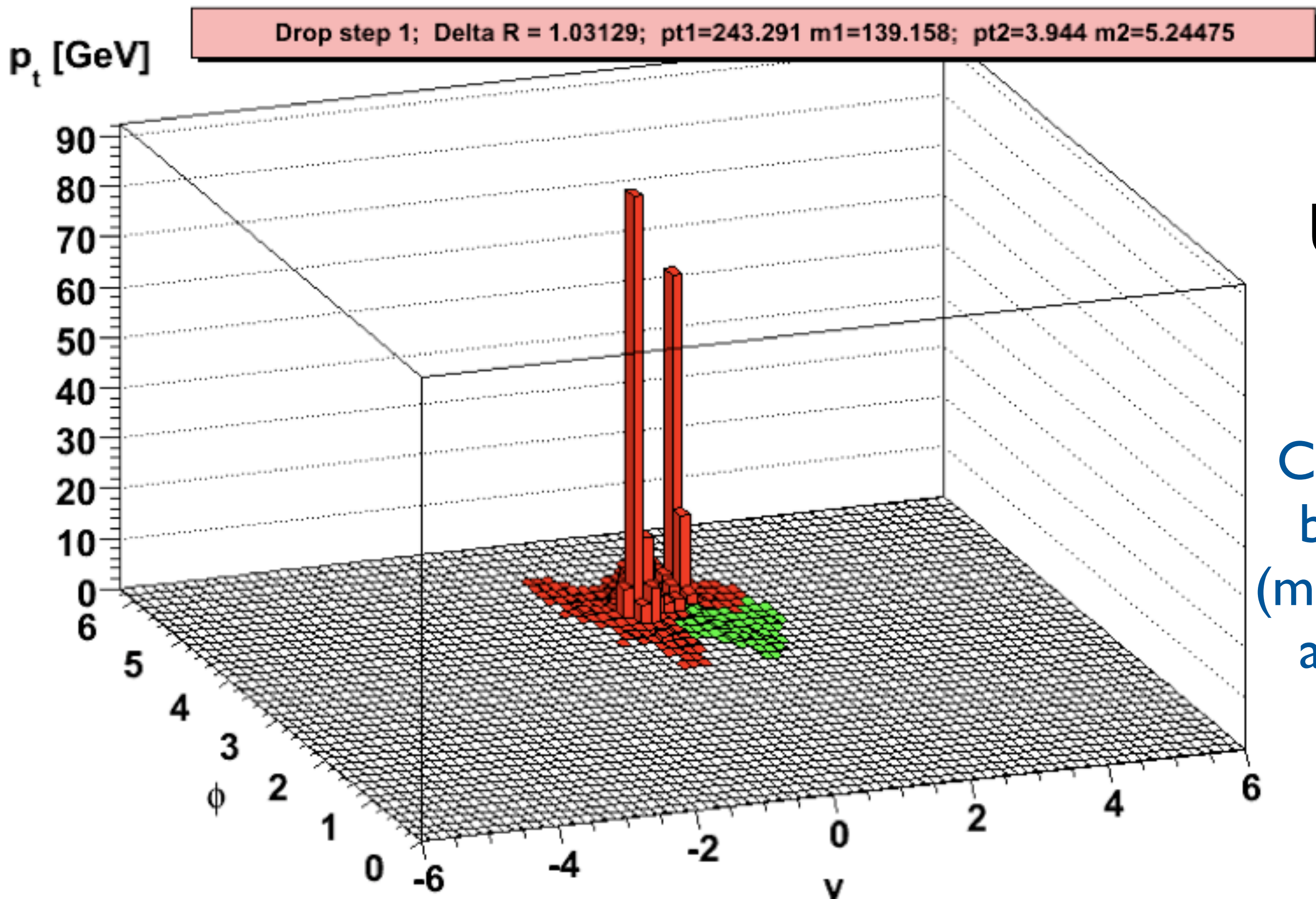
Start with the hardest jet

Use C/A with large R=1.2

m_j = 150 GeV

Boosted Higgs tagger

pp →ZH → ννbb



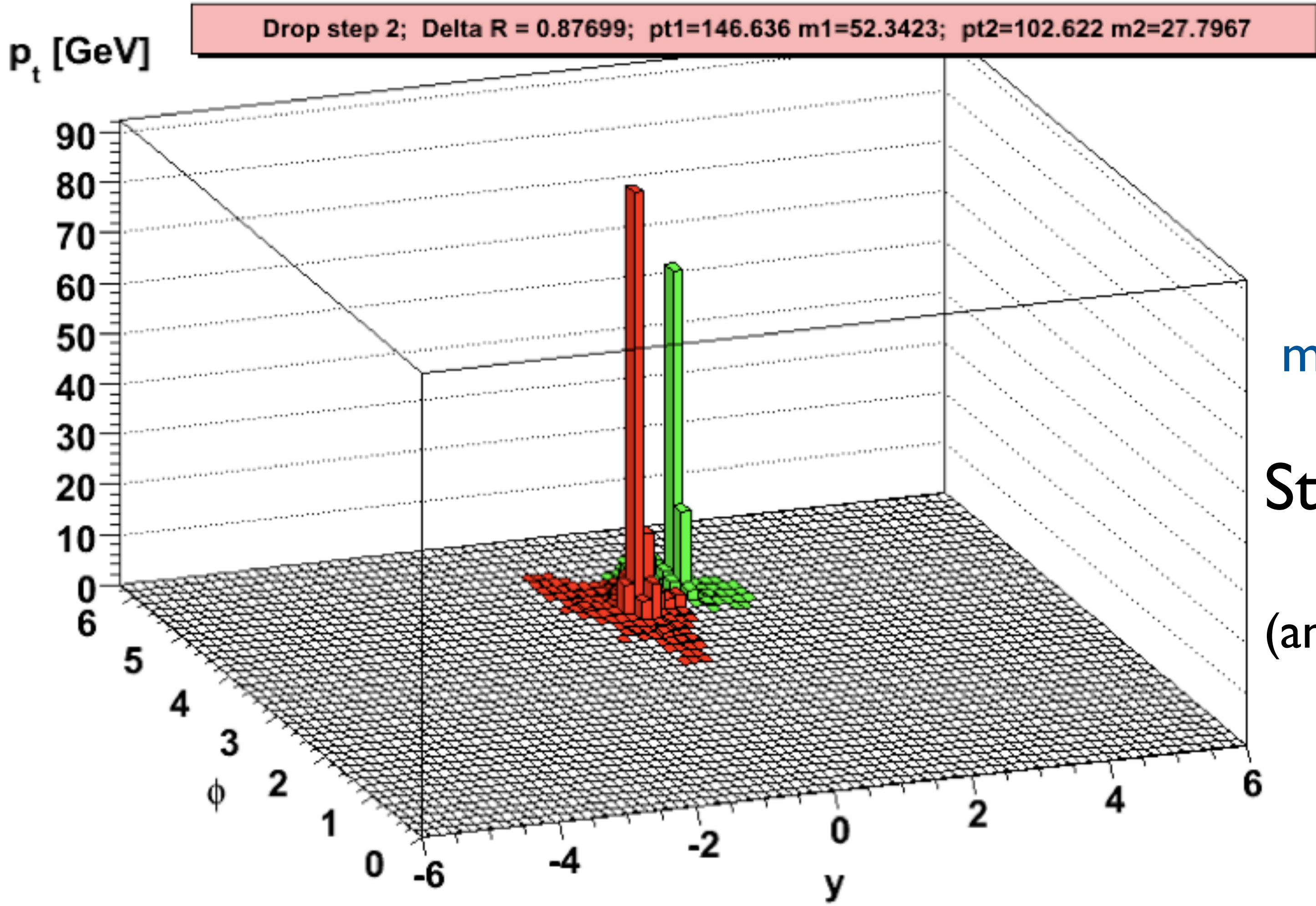
Undo last step of clustering

Check how the mass splits between the two subjets (m_1 = 139 GeV, m_2 = 5 GeV) and how asymmetric the splitting is

If $\frac{max(m_1, m_2)}{m_j} > \mu$ or $\frac{min(p_{t1}^2, p_{t2}^2)}{m_j^2}\Delta R_{12}^2 < y_{cut}$ repeat

Boosted Higgs tagger

$pp \to ZH \to \nu\nu bb$



m_1 = 52 GeV, m_2 = 28 GeV

Stop when a large mass drop is observed

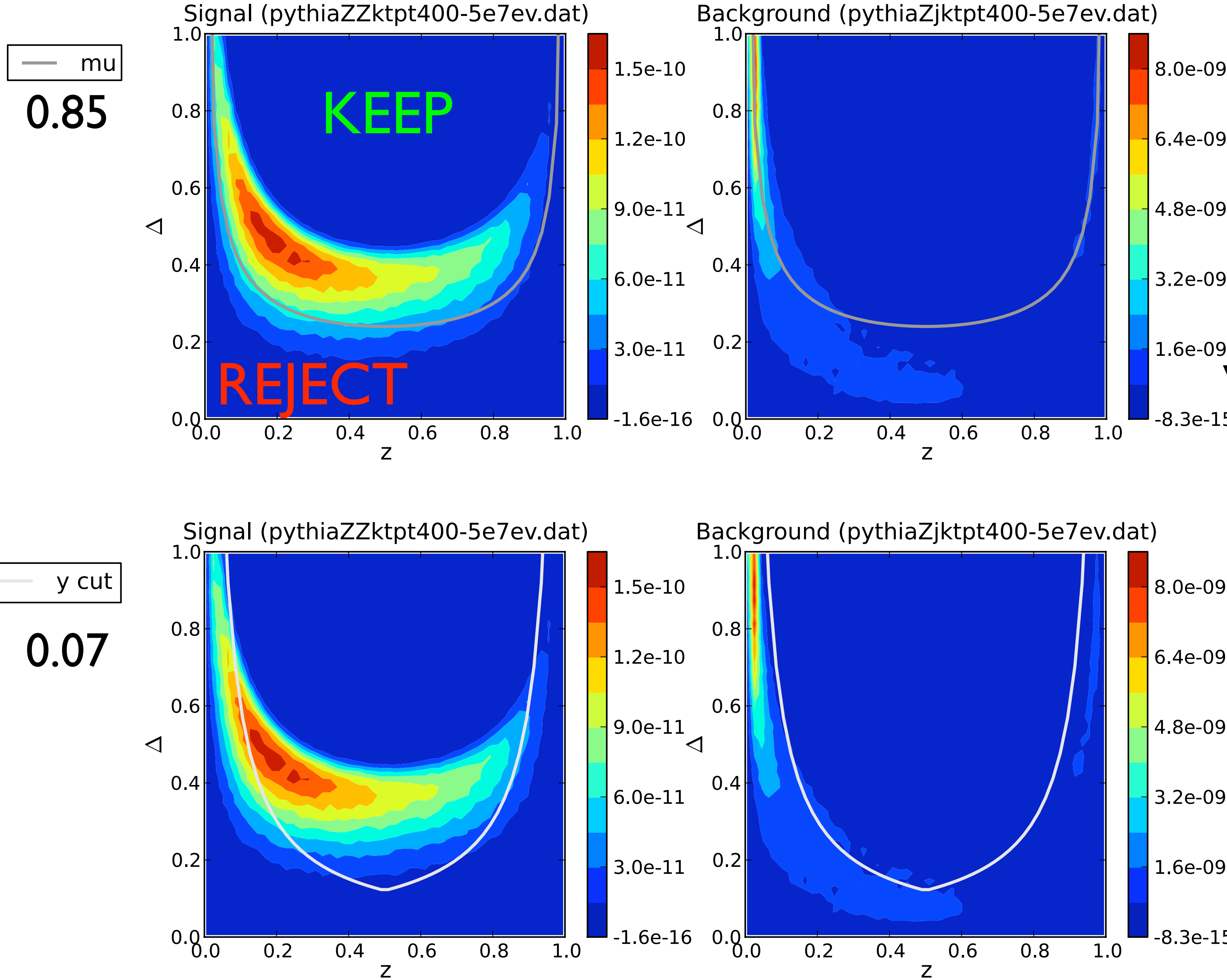
(and recombine these two jets)

[NB. Parameters used μ = 0.67 and y_{cut} = 0.09]

Understand the cuts

ZZ ('signal') Zj ('background')

P. Quiroga
S. Sapeta
(preliminary)



Values of z and ΔR_{ij} at the 'relevant splitting' (i.e. the one satisfying the given conditions)

Jet substructure as filter

The ***jet substructure***
can be exploited to help **removing contamination**
from a soft background

- Jet 'filtering' — Butterworth, Davison, Rubin, Salam, 2008
  Break jet into subjets at distance scale R_{filt}, retain n_{filt} hardest subjets
- Jet 'trimming' — Krohn, Thaler, Wang, 2009
  Break jet into subjets at distance scale R_{trim}, retain subjets with $p_{t,subjet} > \varepsilon_{trim}\, p_{t,jet}$
- Jet 'pruning' — S. Ellis, Vermilion, Walsh, 2009
  While building up the jet, discard softer subjets when $\Delta R > R_{prune}$ and $\min(p_{t1},p_{t2}) < \varepsilon_{prune}\,(p_{t1}+p_{t2})$

Aim: limit sensitivity to background while retaining bulk of perturbative radiation

(Filtering, trimming and pruning are in the end effectively quite similar)

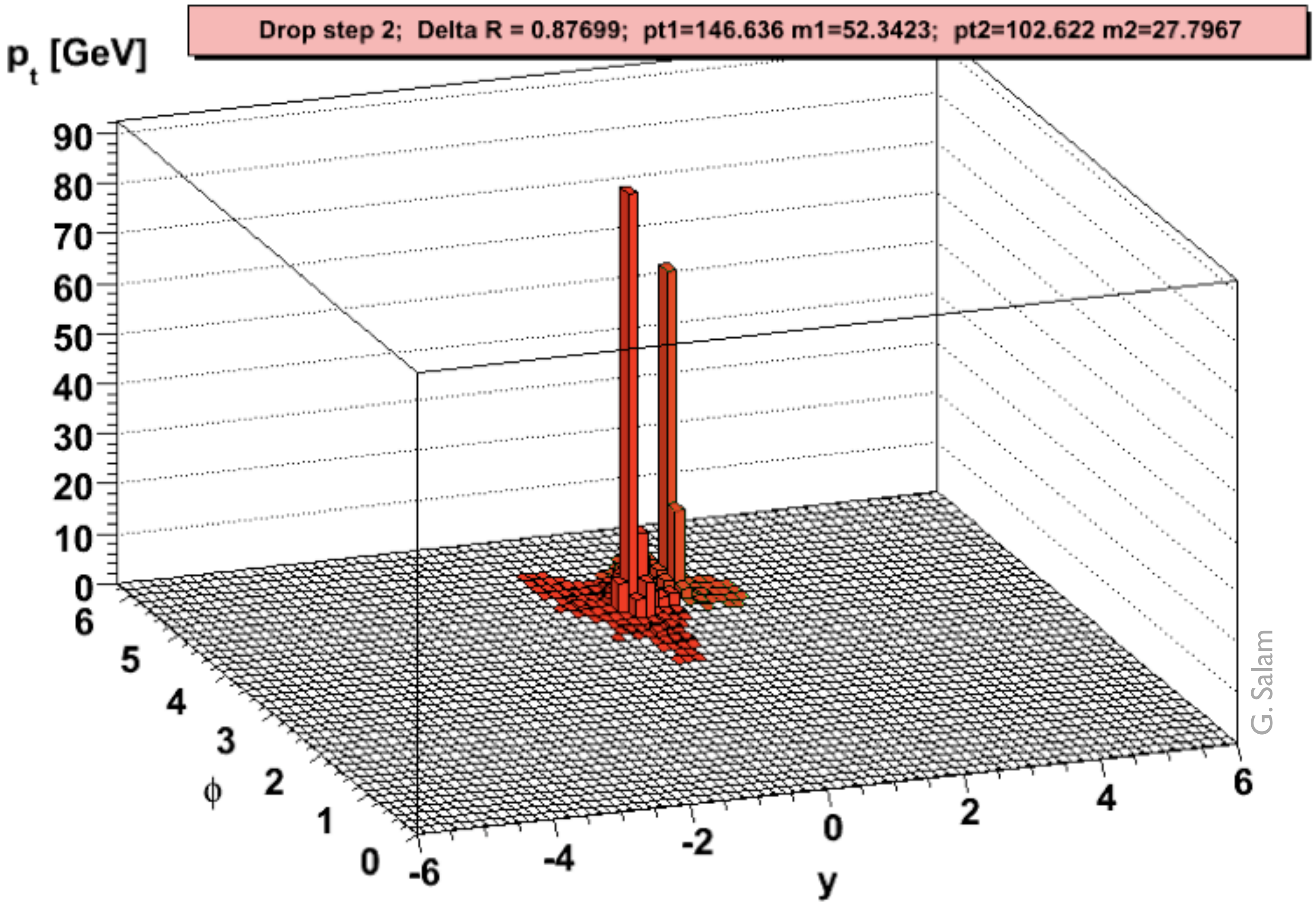
Cambridge/Aachen with filtering

Butterworth, Davison, Rubin, Salam, arXiv:0802.2470

- Cluster with C/A and a given R
- Undo the clustering of each jet down to subjets with radius $x_{filt}R$
- Retain only the n_{filt} hardest subjets

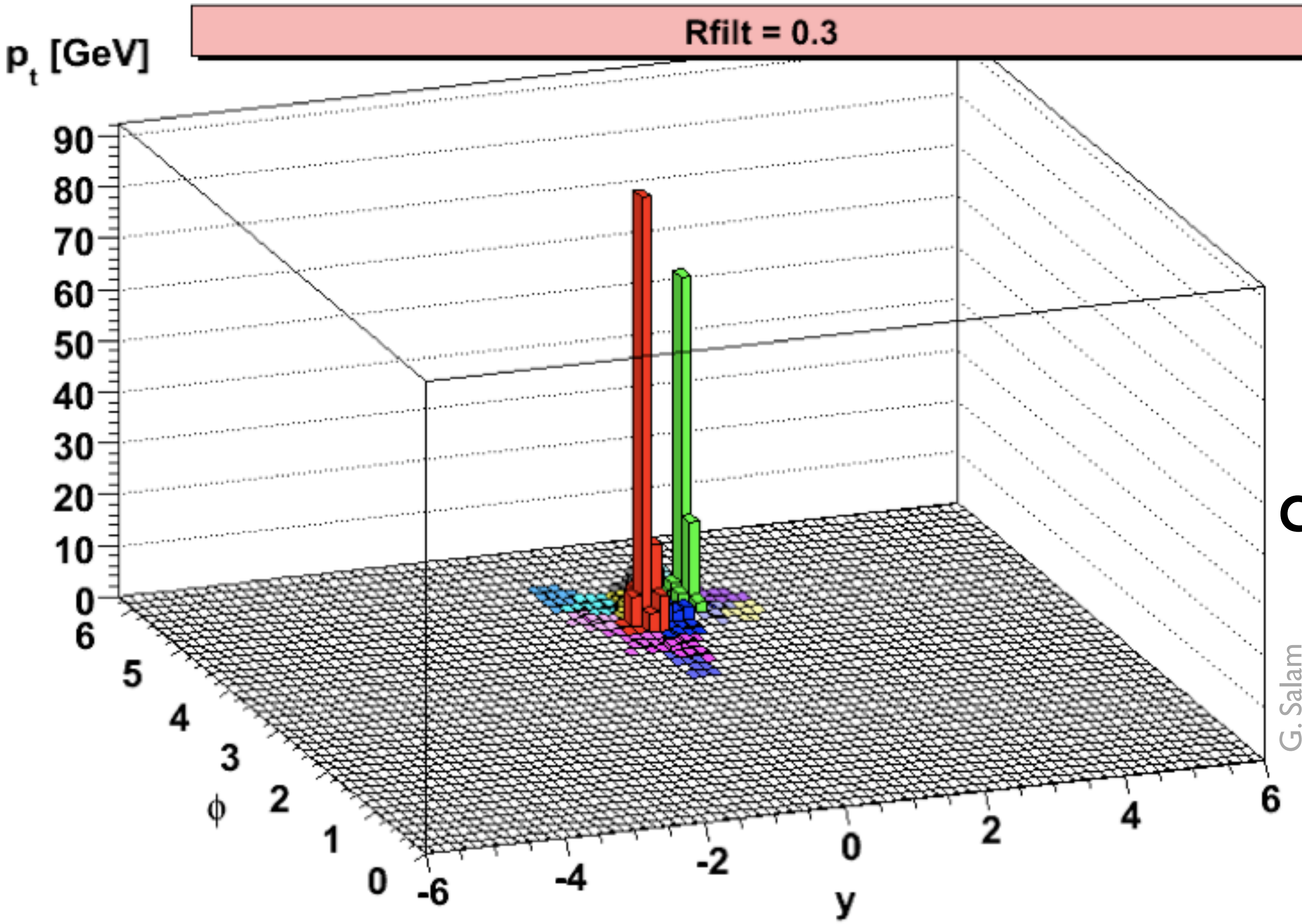
Filtering in action

Butterworth, Davison, Rubin, Salam, arXiv:0802.2470



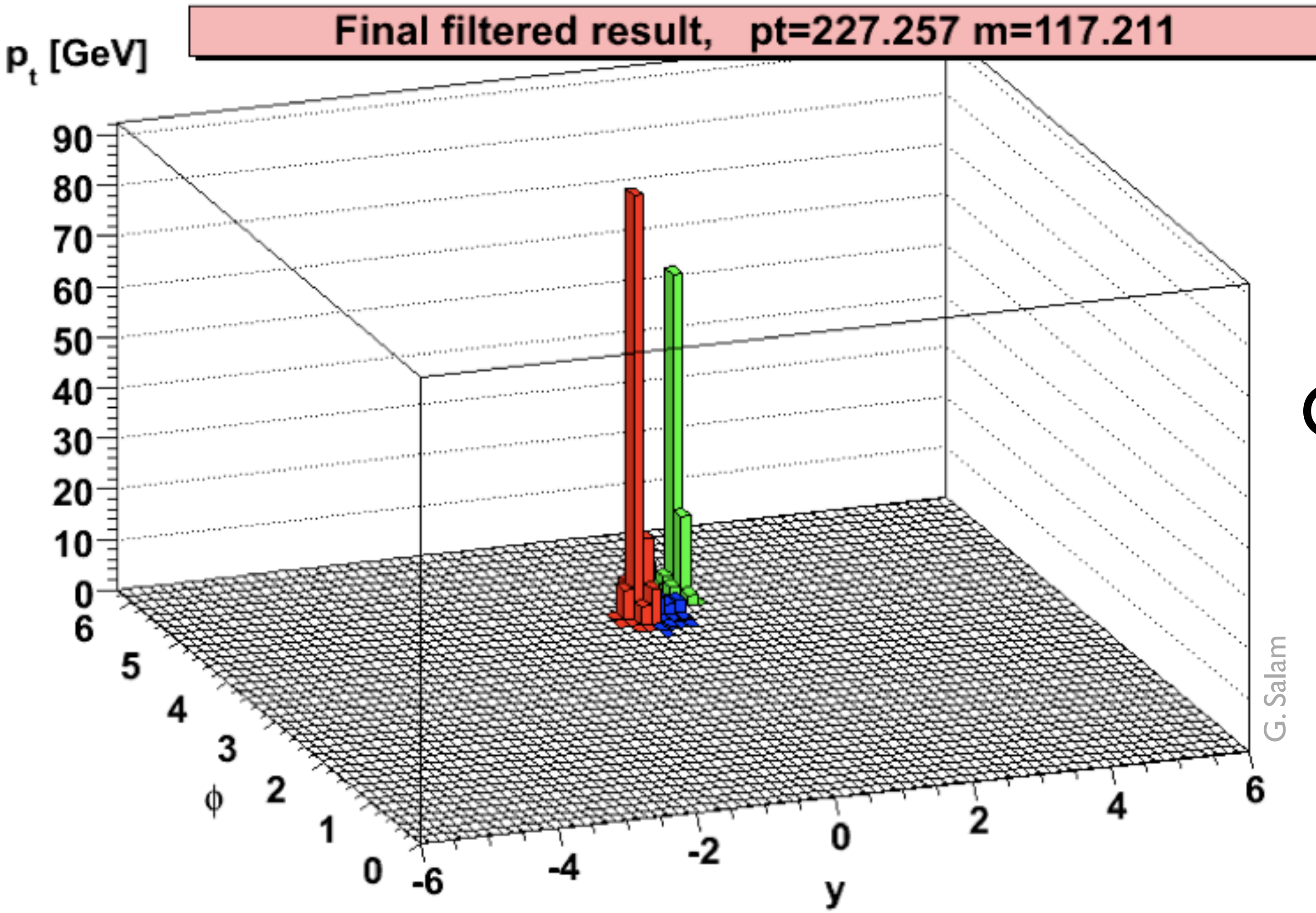
Start with a jet

Filtering in action



Recluster the contituents with R_{filt}

Filtering in action



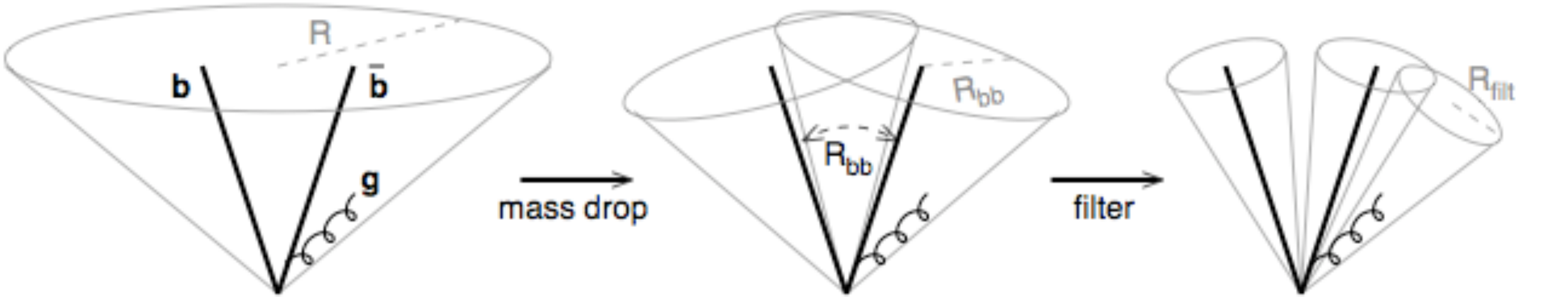
Only keep the n_{filt} hardest jets

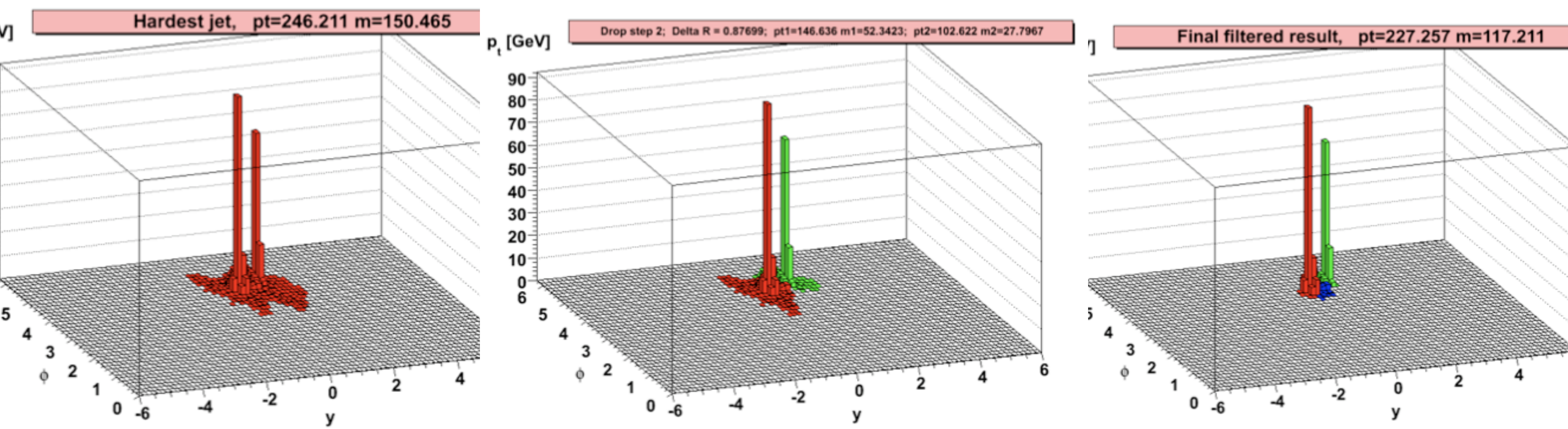
The low-momentum stuff surrounding the hard particles has been removed

Boosted Higgs analysis

$pp \to ZH \to \nu\bar{\nu}b\bar{b}$

Butterworth, Davison, Rubin, Salam, 2008





Cluster with a large R

Undo the clustering into subjets, until a large mass drop is observed

Re-cluster with smaller R, and keep only 3 hardest jets