iSTEP Summer School
Tsinghua University - Beijing
July 2016

Introduction to FastJet tutorials

Matteo Cacciari
LPTHE Paris
Université Paris Diderot

Includes material from
Gavin Salam and Grégory Soyez

Prerequisites

- FastJet : http://fastjet.fr

- Optional :

  - Gnuplot (for plotting) http://gnuplot.info

  - FastJet contribs : http://fastjet.hepforge.org/contrib/

All available on the MC4BSM VM, except the contribs

Purpose of FastJet

C++ library for performing fast jet clustering and (some) jet manipulation and analysis (background subtraction, jet substructure, ...)

Used by all LHC experiments and many phenomenologists

Interfaced and wrapped by many other packages (Pythia, RIVET, Delphes, ...)

FastJet distribution

```
> wget http://fastjet.fr/repo/fastjet-3.2.0.tar.gz
> tar zxf fastjet-3.2.0.tar.gz
```

```
14:39 bogon:tmp>ls fastjet-3.2.0/
AUTHORS                 depcomp*
BUGS                    doc/
COPYING                 example/
ChangeLog               fastjet-config.in*
Doxyfile                fortran_wrapper/
INSTALL                 include/
Makefile.am             install-sh*
Makefile.in             ltmain.sh
NEWS                    m4/
README                  makefile.static
TODO                    missing*
aclocal.m4              plugins/
config.guess*           src/
config.h.in             test-compare.sh*
config.sub*             test-script-output-orig.txt
configure*              test-static.sh*
configure.ac            tools/
```

FastJet quick-start

Go to http://fastjet.fr/quickstart.html and follow the instructions for a very basic introduction

Go to the examples/ directory for a set of examples with various functionalities of FastJet

Jet clustering in FastJet

```
/// define a jet definition
JetDefinition jet_def(JetAlgorithm jet_algorithm,
                      double R,
                      RecombinationScheme rec_sch = E_scheme);
```

jet_algorithm can be any one of the four IRC safe algorithms, or also most of the old IRC-unsafe ones, for legacy purposes

```
/// create a ClusterSequence, extract the jets
ClusterSequence cs(input_particles, jet_def);
vector<PseudoJet> jets = sorted_by_pt(cs.inclusive(jets));
...
// pt of hardest jet
double pt_hardest = jets[0].pt();
...
// constituents of hardest jet
vector<PseudoJet> constits = jets[0].constituents();
```

(Boosted) jet studies at the LHC

Lily Asquith, summary talk at BOOST 2015

Boost is about:

1. Tagging high pT objects (SM and BSM)
2. Improving measurements (pileup, mass resolution etc)

ATLAS and CMS have taken different approaches to these things from day one.

ATLAS:
AKT4 CA12 split-fitered (BDRS)
AKT10 trimmed (R3/R2)
N-subjettiness WTA
JVT/ ρ
D2

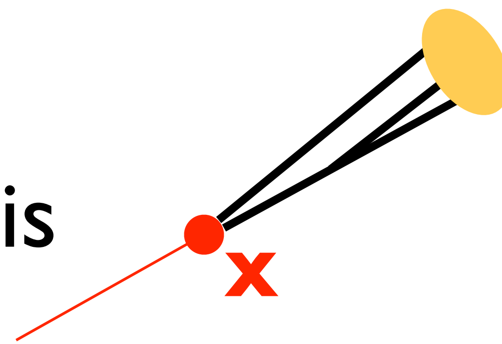
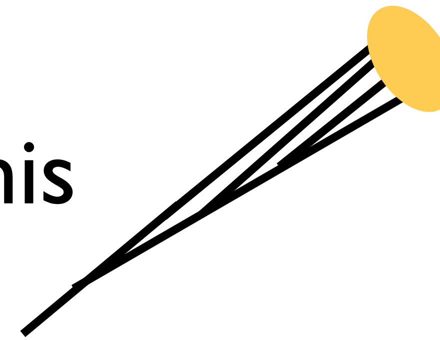
CMS:
AKT5 CA8 pruned (p510)
CA15 HTT
N-subjettiness one-pass
Puppi
Soft drop

Essentially none of these tools existed
as lately as seven years ago

Glossary

What	i.e.	When	Ref.
AKT	Anti-kt algorithm	2008	0802.1189
CA	Cambridge/Aachen algorithm	1999	9907280
BDRS	mass-drop tagger, includes filtering	2008	0802.2470
trimmed	Trimming, tagger/groomer	2009	0912.1342
pruned	Pruning, tagger/groomer	2009	0903.5081
HTT	HepTopTagger	2009	0910.5472
N-subjettiness	jet shape function, used in tagging	2010	1011.2268
WTA	Winner-Take-All (recombination scheme)	2013	1310.7584
one-pass	choice of axis for N-subjettiness	2010	
JVT	Jet Vertex Tagger (used in pileup subtr.)	2014	
ρ	background density (used in pileup subtr.)	2007	0707.1378
D2	jet shape function, used in tagging	2014	1409.6298
PUPPI	particle-by-particle pileup subtr.	2014	1407.6013
Soft Drop	tagger/groomer	2014	1402.2657

Tagging

How to tell this  from this  ?

Decay of a heavy (boosted) object

Light parton fragmentation

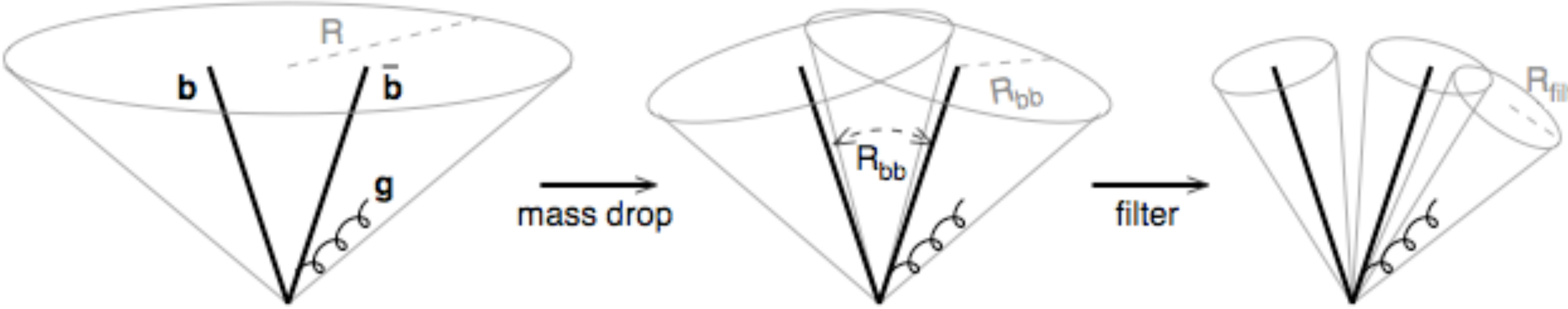
Tagging and Grooming

- The substructure of a jet can be exploited to
  - ***tag*** *a particular structure inside the jet, i.e. a massive particle*
    - *First examples: Higgs (2-prong decay), top (3-prong decay)*
  - *remove background contamination from the jet or its components, while keeping the bulk of the perturbative radiation (often generically denoted as* ***grooming****)*
    - *First examples: filtering, trimming, pruning*

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

The BDRS tagger/groomer

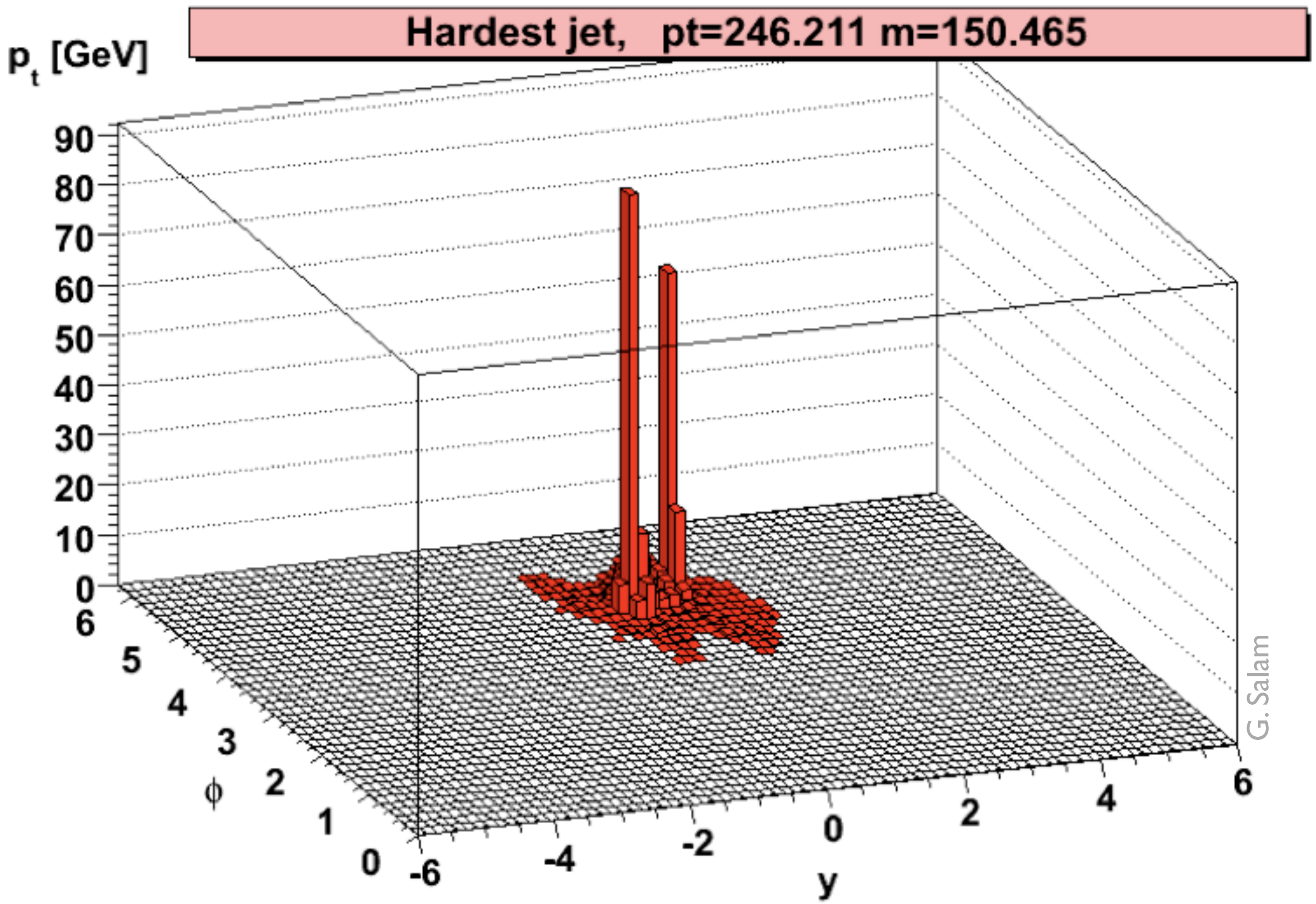
Butterworth, Davison, Rubin, Salam, 2008



- A two-prong tagger/groomer for boosted Higgs, which
  - Uses the **Cambridge/Aachen** algorithm (because it's 'physical')
  - 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 ('**grooming**')

BDRS: tagging

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



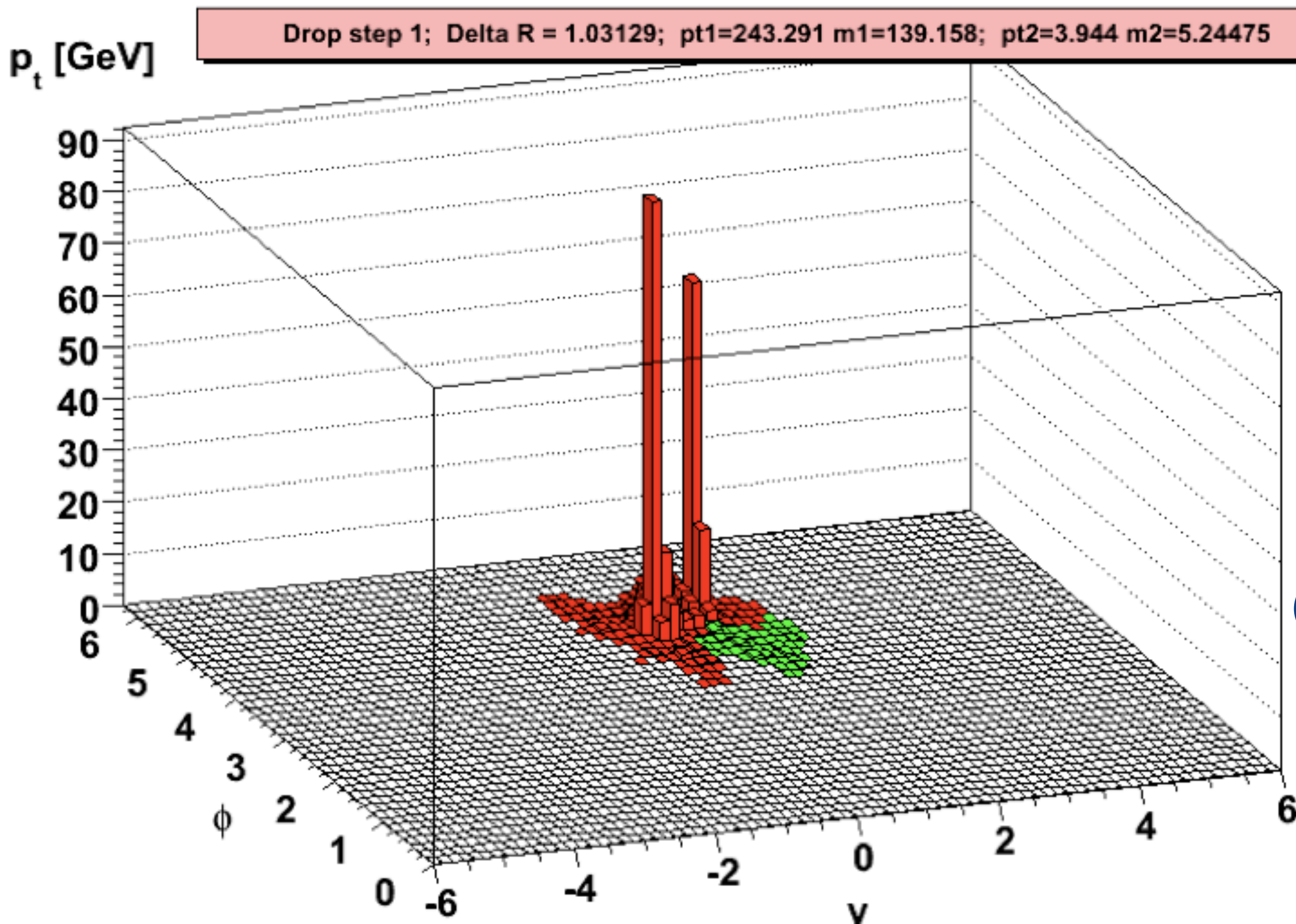
Start with the hardest jet

Use C/A with large R=1.2

m_j = 150 GeV

BDRS: tagging

pp → ZH → ννbb

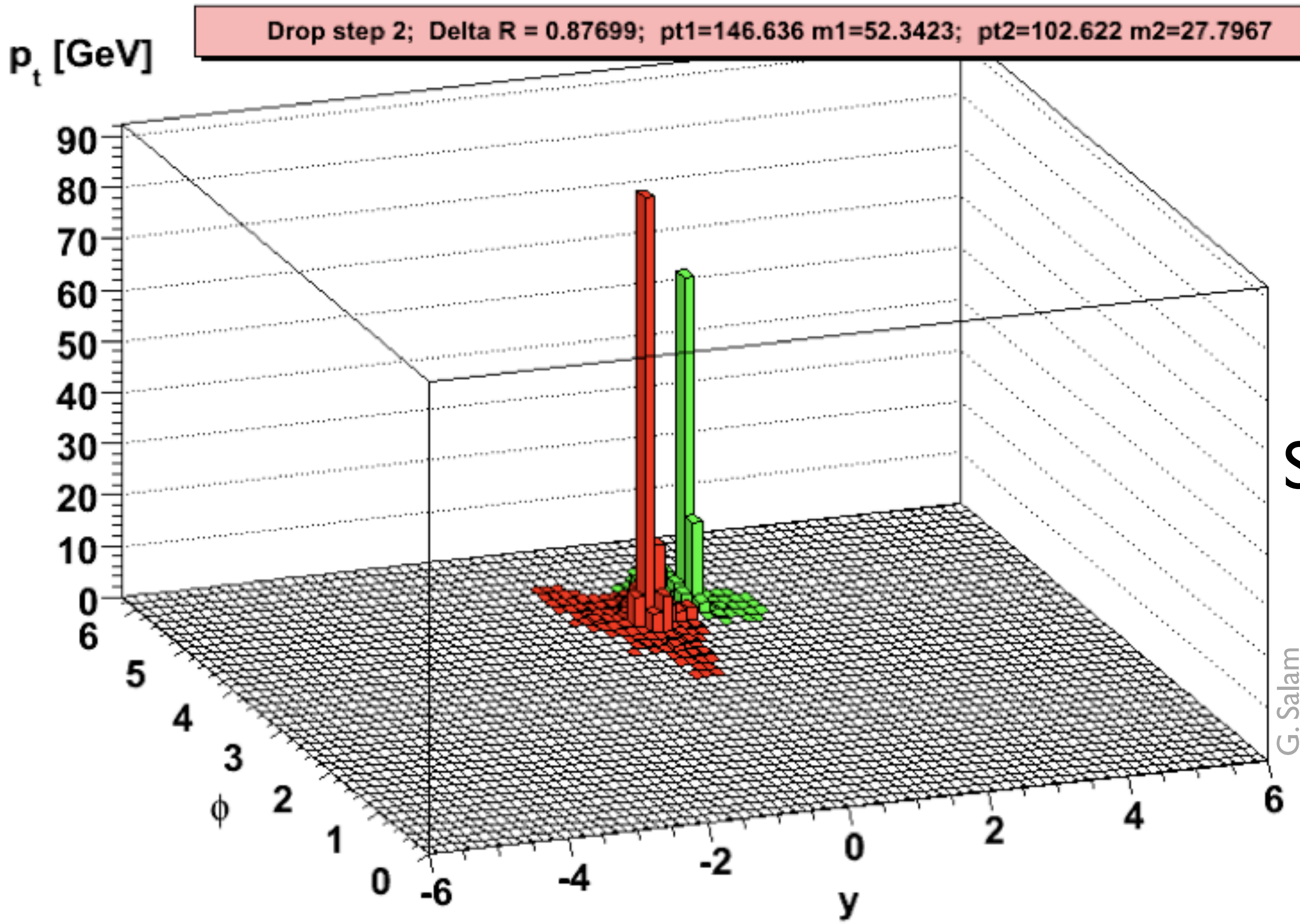


Undo last step of clustering

Check how the mass splits between the two subjects (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

pp → ZH → ννbb

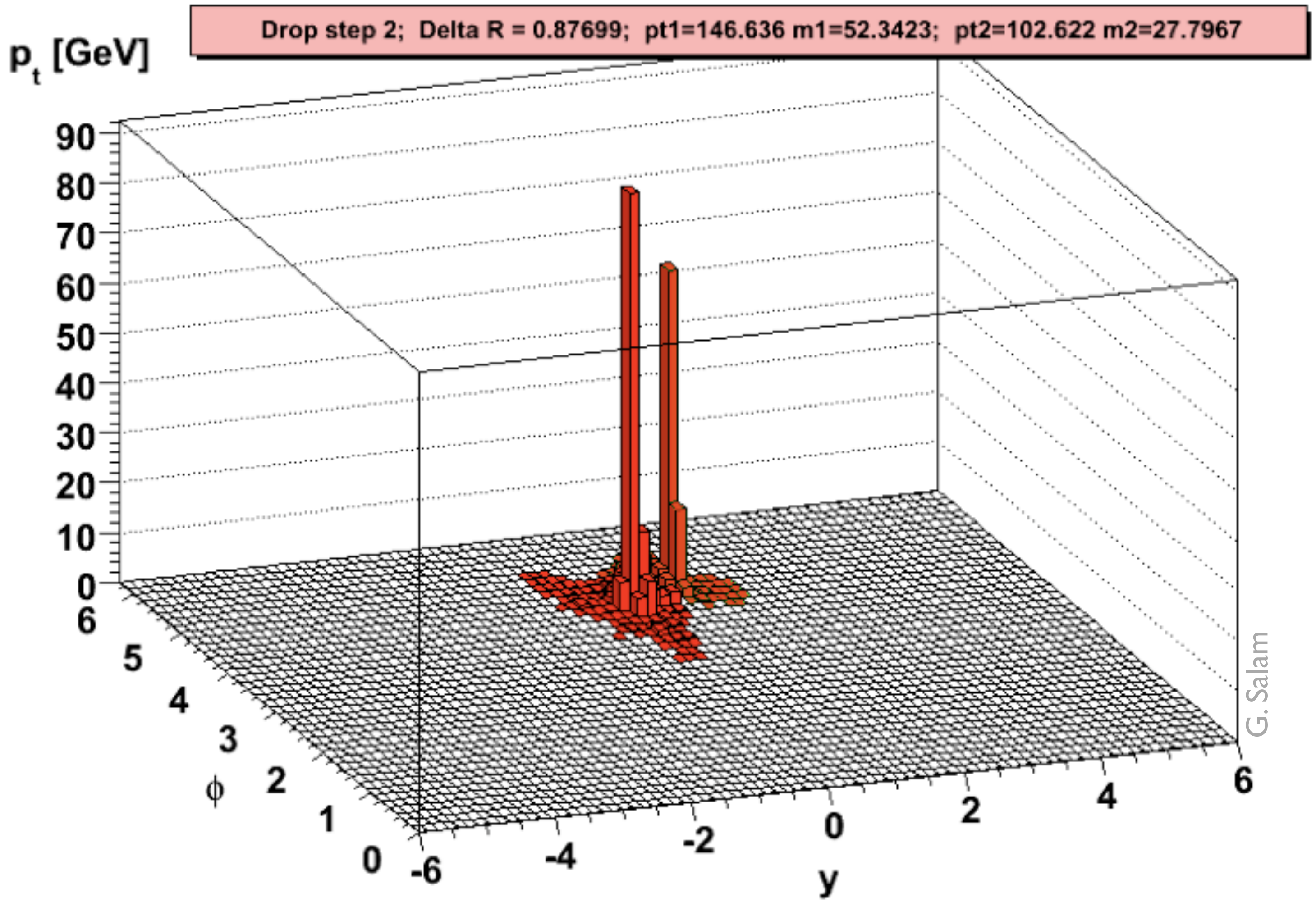


m_1 = 52 GeV, m_2 = 28 GeV

Stop when a large mass drop is observed (and **recombine** these two jets)

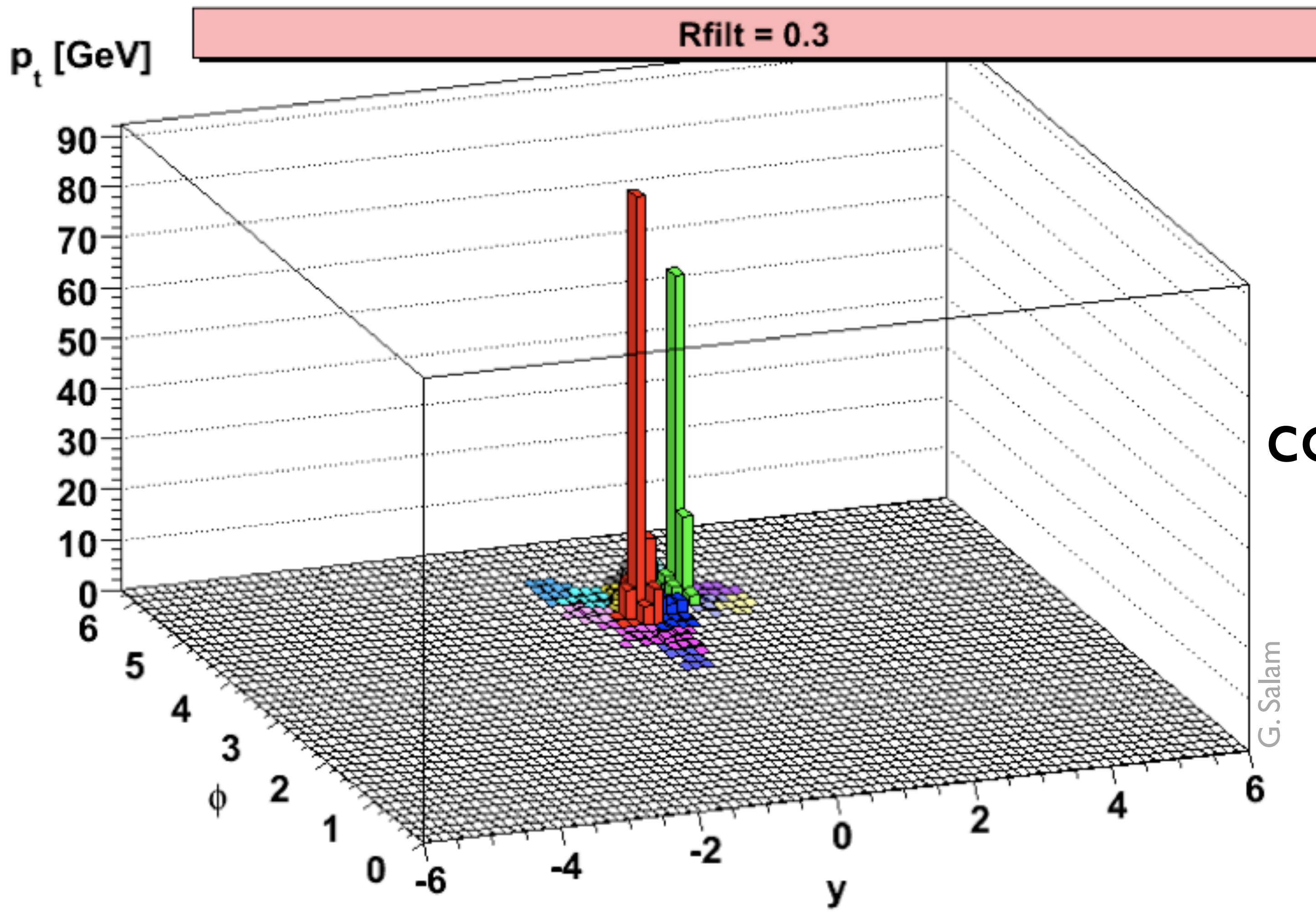
[NB. Parameters used μ = 0.67 and y_{cut} = 0.09]

pp → ZH → ννbb



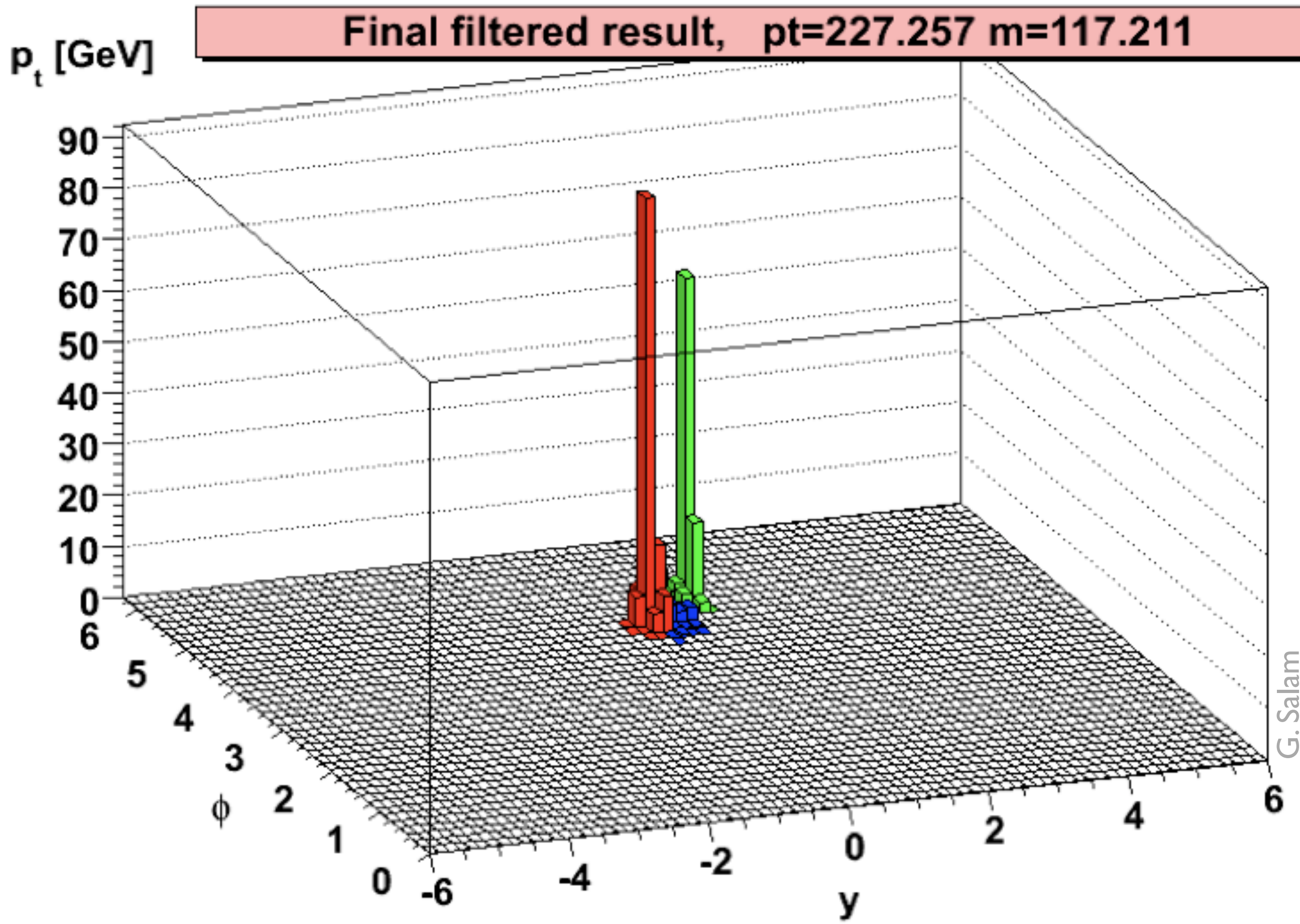
Start with the recombined jet

pp →ZH → ννbb



Recluster the contituents with R_{filt}

pp → ZH → ννbb



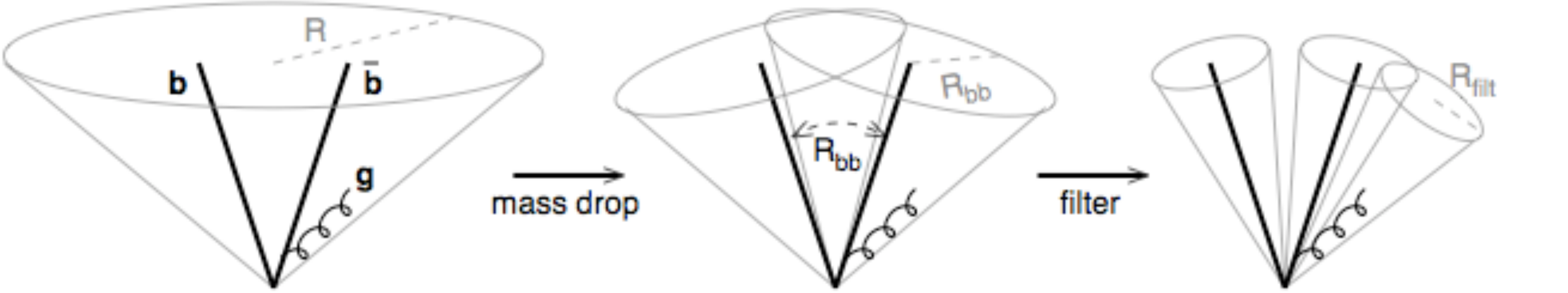
Only keep the n_{filt} hardest jets

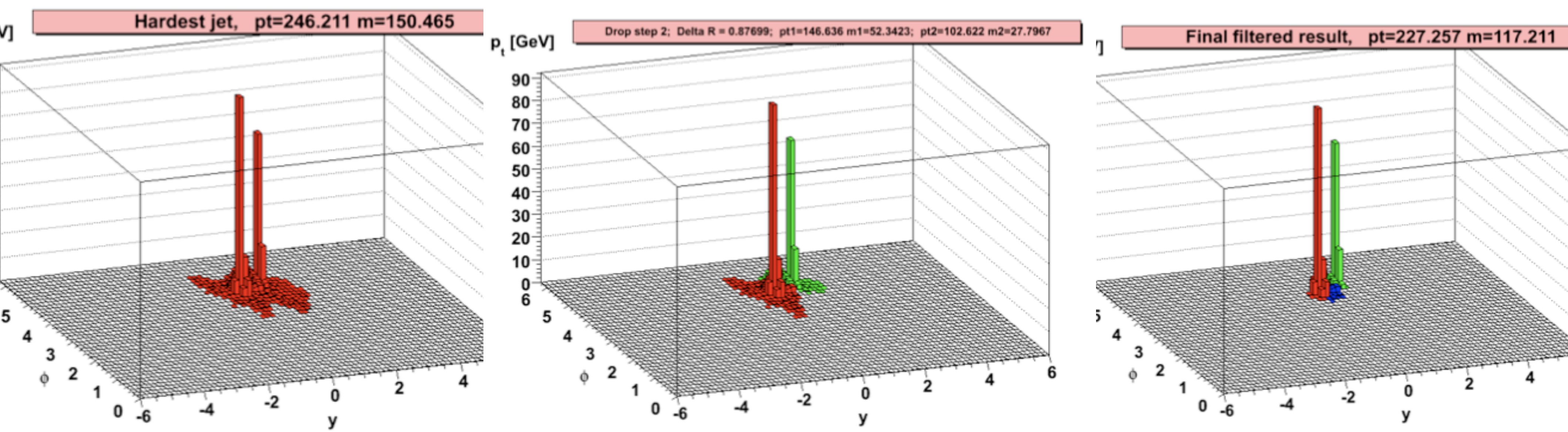
The low-momentum stuff surrounding the hard particles has been removed

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

Visualisation of BDRS

Butterworth, Davison, Rubin, Salam, 2008





Cluster with a large R

Undo the clustering into subjets, until a large asymmetry/mass drop is observed: tagging step

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

BDRS in FastJet

In FastJet

```
#include “fastjet/tools/MassDropTagger.hh”
#include “fastjet/tools/Filter.hh”

JetDefinition jet_def(cambridge_algorithm, 1.2);
ClusterSequence cs(input_particles, jet_def);

// define the tagger and use it
MassDropTagger md_tagger(0.667, 0.09);
PseudoJet tagged = md_tagger(jets[0]);

// define the filter and use it
Filter filter(0.3,SelectorNHardest(3));
Pseudojet higgs = filter(tagged);     // this is the Higgs!!
```

The real analysis is slightly more refined (b-tagging, dynamical filter radius, etc) but the main features are already present here

BDRS in boosted-HZ tutorial

The BDRS technique can be used in the **boosted-HZ tutorial** that you have been given

Alternatives to hierarchical substruct.

- If what we are interested in is the structure of the constituents of a jet, the "jet" itself is not the most important feature.
- A different algorithm, or simply the study of the constituents in a certain patch will also do. Selected alternatives are:
  - Use of jet-shapes to characterise certain features
    - e.g. *N-subjettiness*: how many subjets a jets appears to have

      Thaler, van Tilburg, 2011
  - Alternative ways of clustering
    - e.g. *Qjets*: the clustering history not deterministic, but controlled by random probabilities of merging. Can be combined with, e.g. pruning

      Ellis, Hornig, Roy, Krohn, Schwartz, 2012
  - Use information from matrix element
    - e.g. *shower deconstruction*: use analytic shower calculations to estimate probability that a certain configuration comes from signal or from background

      Soper, Spannowsky, 2011
  - Use event shapes mimicking jet properties
    - e.g. *JetsWithoutJets*, mimicking trimming

      Bertolini, Chen, Thaler, 2013

N-subjettiness

Thaler, van Tilburg, 2010

$$\tau_N^{(\beta)} = \sum_i p_{Ti} \min \left\{ R_{1,i}^{\beta}, R_{2,i}^{\beta}, \ldots, R_{N,i}^{\beta} \right\}$$

Sum over constituents of a jet

Distances to axes of N subjets

τ_N measures departure from N-parton energy flow:

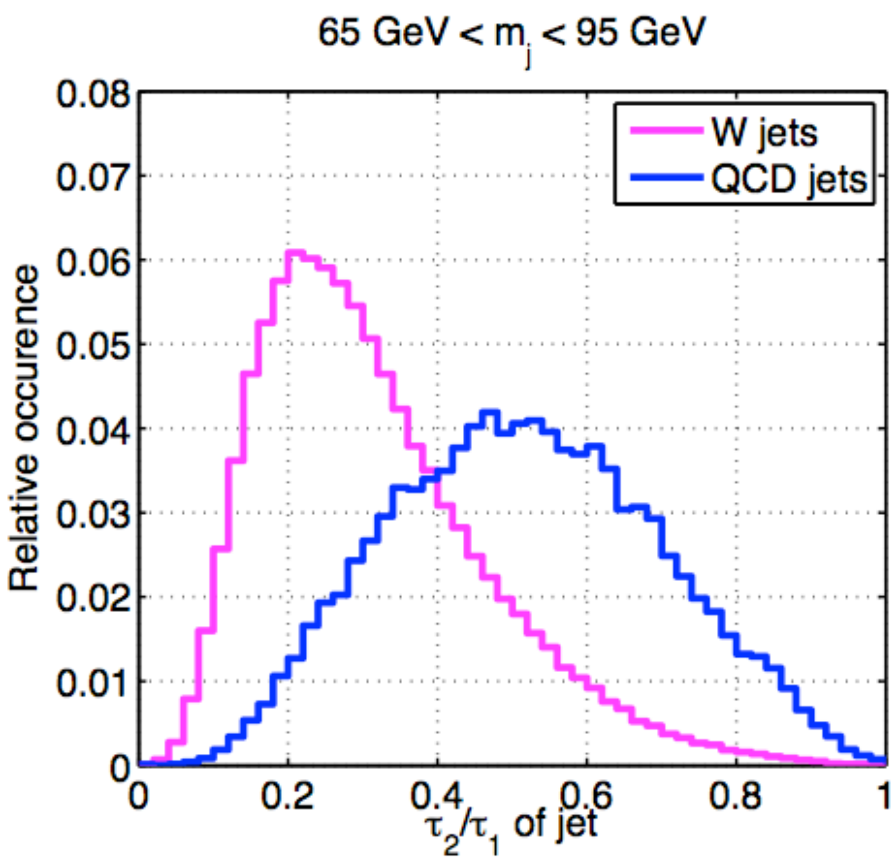
if a jet has N subjets, τ_{N-1} should be much larger than τ_N

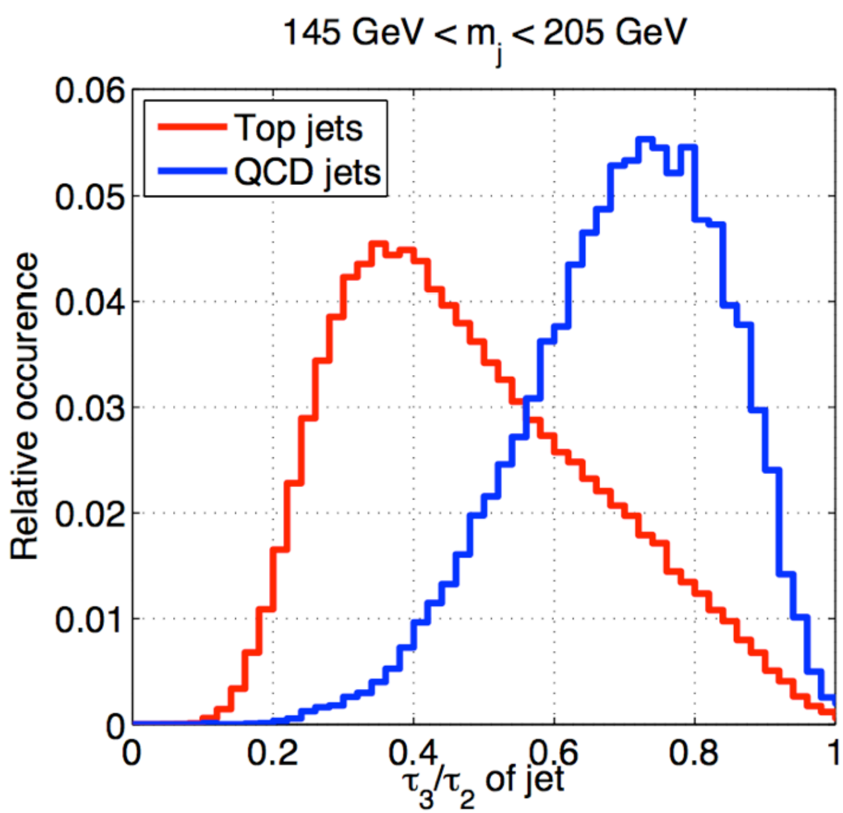
N-subjettiness

Thaler, van Tilburg, 2010

$$\tau_{N,N-1}^{(\beta)} \equiv \frac{\tau_N^{(\beta)}}{\tau_{N-1}^{(\beta)}}$$

A jet with a **small** $\tau_{N,N-1}$ is more likely to have N than N-1 subjets





(from 1011.2268, with β=1)

Nsubjettiness in FastJet

Important: first download and install FastJet contribs

Also, have in the Makefile

```
INCLUDE += `fastjet-config --cxxflags`/contrib
LIBRARIES += -lNsubjettiness
```

In FastJet

```
#include “fastjet/contrib/Nsubjettiness.hh”

double beta = 1.0;
double tau21_cut = 0.3;
NsubjettinessRatio nSub21_beta1(2,1,
                                OnePass_WTA_KT_Axes(),
                                UnnormalizedMeasure(beta));
double tau21_beta1 = nSub21_beta1(jets[0]);
if ( tau21_beta1 > tau21_cut ) jets[0]=PseudoJet();
```

Tutorial's README

Instructions are in the file **boosted-HZ-tutorial.pdf**

Prerequisites are a working installation of **FastJet** (http://fastjet.fr)

A working installation of **Gnuplot** will also be useful, as a plotting macro (plot.gp) is provided

The following event samples (NB: about 80MB each) are needed for this tutorial:

- dijet production (pt > 500 GeV) (background only):
http://www.lpthe.jussieu.fr/~cacciari/public/fastjet/events/pythia8-10000dijets-ptmin500.UW.gz

- HZ and dijet production (pt > 500 GeV) (signal + background):
http://www.lpthe.jussieu.fr/~cacciari/public/fastjet/events/pythia8-200HZ-10000dijets-ptmin500.UW.gz

Backup

Soft Drop declustering

Larkoski, Marzani, Soyez, Thaler, 2014

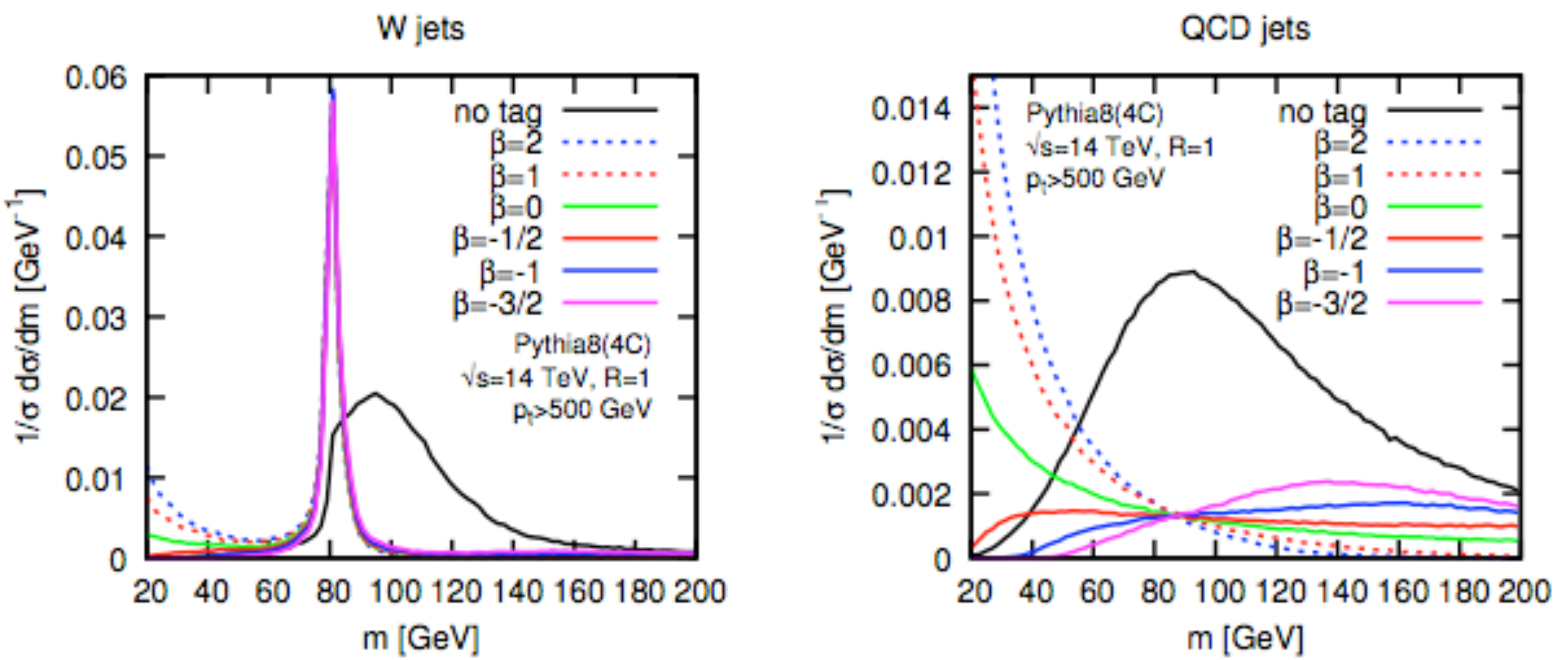
Decluster and drop softer constituent unless

$$\text{Soft Drop Condition:} \quad \frac{\min(p_{T1}, p_{T2})}{p_{T1} + p_{T2}} > z_{\text{cut}} \left(\frac{\Delta R_{12}}{R_0} \right)^{\beta}$$

i.e. remove wide-angle soft radiation from a jet

The paper contains

- ✓ analytical calculations and comparisons to Monte Carlos
- ✓ study of effect of non-perturbative corrections
- ✓ performance studies



Example of SoftDrop performance when used as a boosted W tagger

Larkoski, Salam, Thaler 2013

Energy correlation functions

Probes of N-prong structures without requiring identification of subjets

$$\mathrm{ECF}(N,\beta) = \sum_{i_1<i_2<\ldots<i_N\in J} \left(\prod_{a=1}^{N} p_{Ti_a}\right)\left(\prod_{b=1}^{N-1}\prod_{c=b+1}^{N} R_{i_b i_c}\right)^{\beta}$$

Angular (y-φ) distances between constituents

ECF(N+1) is zero if there are only N particles

More generally, *if there are N subjets one expects ECF(N+1) to be much smaller than ECF(N)*

[because radiation will be mainly soft/collinear to subjets]

Larkoski, Salam, Thaler 2013

Discriminators

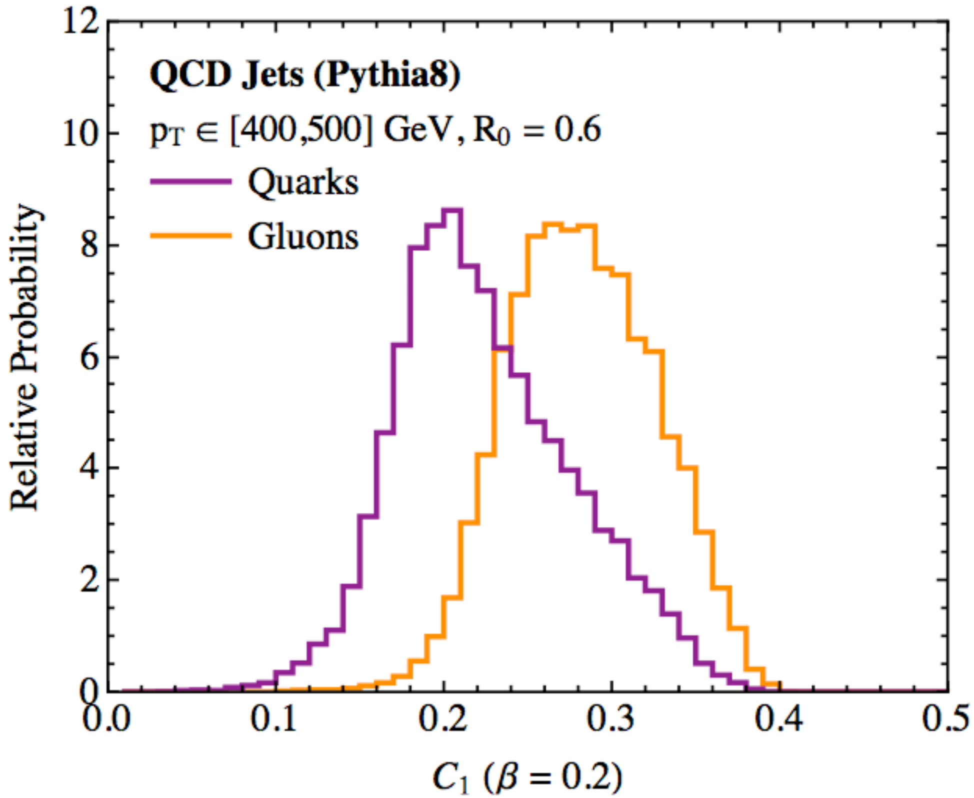
$$r_N^{(\beta)} \equiv \frac{\mathrm{ECF}(N+1,\beta)}{\mathrm{ECF}(N,\beta)}$$

small for N prongs:
if N hard partons, small if radiation
only soft-collinear

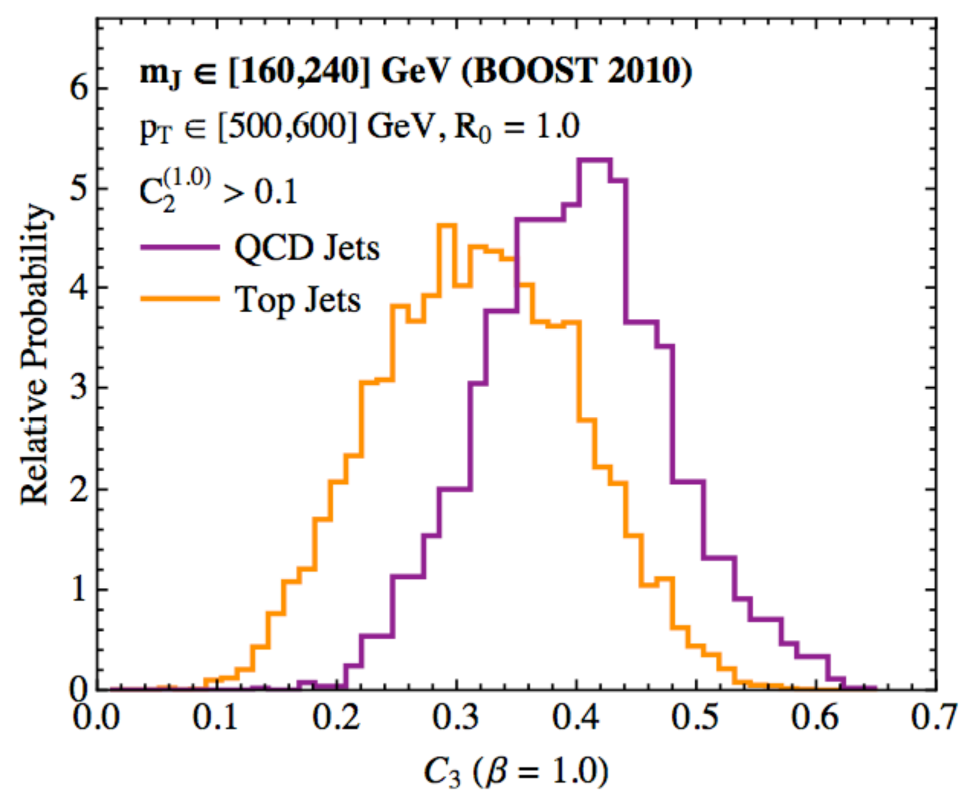
$$C_N^{(\beta)} \equiv \frac{r_N^{(\beta)}}{r_{N-1}^{(\beta)}} = \frac{\mathrm{ECF}(N+1,\beta)\,\mathrm{ECF}(N-1,\beta)}{\mathrm{ECF}(N,\beta)^2}$$

A jet with a **small** C_N is more likely
to have N prongs and at most soft/coll radiation

C_1
quark-gluon discriminator



C_3
top tagging



Note different values of β
(chosen to maximise discriminating power)

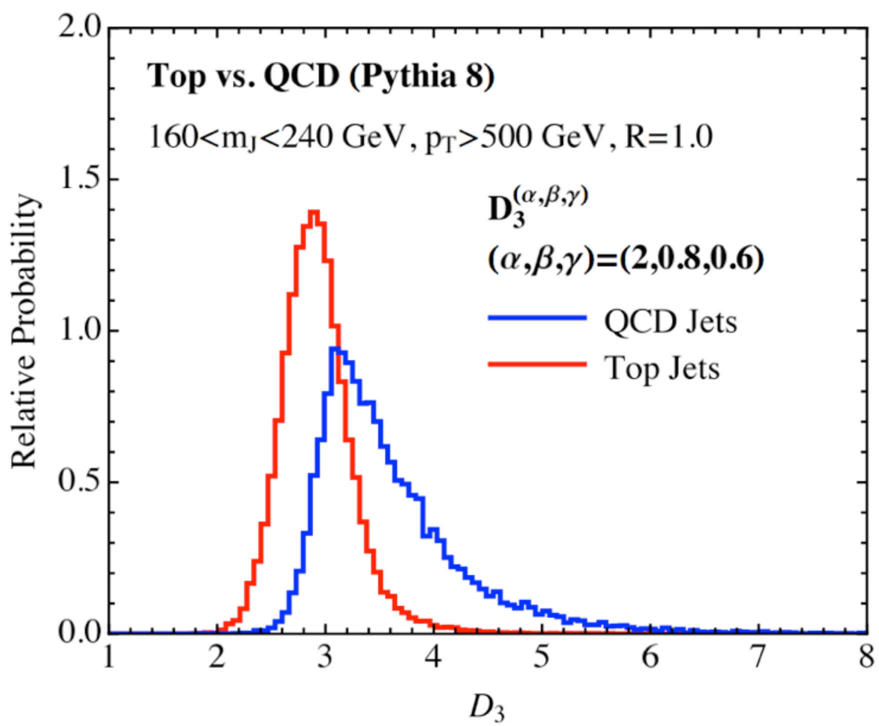
Larkoski, Moult, Neill, 2014

The D functions are variations of the C ones

Instead of $C_2^{(\beta)} = \frac{e_3^{(\beta)}}{(e_2^{(\beta)})^2}$ $\quad C_3^{(\beta)} = \frac{e_4^{(\beta)} e_2^{(\beta)}}{(e_3^{(\beta)})^2}$

define $D_2^{(\beta)} = \frac{e_3^{(\beta)}}{(e_2^{(\beta)})^3}$ $\quad D_3^{(\alpha,\beta,\gamma)} = \frac{e_4^{(\gamma)} \left(e_2^{(\alpha)}\right)^{\frac{3\gamma}{\alpha}}}{\left(e_3^{(\beta)}\right)^{\frac{3\gamma}{\beta}}} + x \frac{e_4^{(\gamma)} \left(e_2^{(\alpha)}\right)^{\frac{2\gamma}{\beta}-1}}{\left(e_3^{(\beta)}\right)^{\frac{2\gamma}{\beta}}} + y \frac{e_4^{(\gamma)} \left(e_2^{(\alpha)}\right)^{\frac{2\beta}{\alpha}-\frac{\gamma}{\alpha}}}{\left(e_3^{(\beta)}\right)^{2}}$

Attempt to improve the discriminating power, and to account for different regions of phase space of radiation
[also, gives an idea of increasing 'sophistication', or complexification]



Jet trimming

Krohn, Thaler, Wang, 2009

1. Cluster all cells/tracks into jets using any clustering algorithm. The resulting jets are called the seed jets.

2. Within each seed jet, recluster the constituents using a (possibly different) jet algorithm into subjets with a characteristic radius $R_{\rm sub}$ smaller than that of the seed jet.

3. Consider each subjet, and discard the contributions of subjet i to the associated seed jet if $p_{Ti} < f_{\rm cut} \cdot \Lambda_{\rm hard}$, where $f_{\rm cut}$ is a fixed dimensionless parameter, and $\Lambda_{\rm hard}$ is some hard scale chosen depending upon the kinematics of the event.

4. Assemble the remaining subjets into the trimmed jet.

Different condition for retaining jets (p_T-cut rather than $n_{\rm filt}$ hardest) with respect to filtering, but otherwise identical

```
#include "fastjet/tools/Filter.hh"

// define trimmer
Filter trimmer(0.3,SelectorPtFractionMin(0.03));
```

Jet pruning

S. Ellis, Vermilion, Walsh, 2009

0. Start with a jet found by any jet algorithm, and collect the objects (such as calorimeter towers) in the jet into a list L. Define parameters $D_{\rm cut}$ and $z_{\rm cut}$ for the pruning procedure.

1. Rerun a jet algorithm on the list L, checking for the following condition in each recombination $i, j \to p$:

$$z = \frac{\min(p_{Ti}, p_{Tj})}{p_{Tp}} < z_{\rm cut} \quad \text{and} \quad \Delta R_{ij} > D_{\rm cut}.$$

   This algorithm must be a recombination algorithm such as the CA or k_T algorithms, and should give a "useful" jet substructure (one where we can meaningfully interpret recombinations in terms of the physics of the jet).

2. If the conditions in 1. are met, do not merge the two branches 1 and 2 into p. Instead, discard the softer branch, i.e., veto on the merging. Proceed with the algorithm.

3. The resulting jet is the *pruned jet*, and can be compared with the jet found in Step 0.

True in general for substructure studies

Exclude soft stuff and large angle recombinations from clustering

Pruning in FastJet

In FastJet

```
#include "fastjet/tools/Pruner.hh"

JetDefinition jet_def(cambridge_algorithm, 1.2);
ClusterSequence cs(input_particles, jet_def);

// define the pruner and use it
double zcut = 0.1;
double rcut_factor = 0.5;

Pruner pruner(cambridge_algorithm, zcut, rcut_factor);

PseudoJet tagged = pruner(jets[0]);
```