The FastJet jet package

Grégory Soyez

Brookhaven National Laboratory

M. Cacciari, G.P. Salam, G. Soyez,

http://www.lpthe.jussieu.fr/~salam/fastjet
**Aim**: Study hard processes

- QCD backgrounds, top quark physics
- Higgs, physics beyond the standard model

**But**: partons are ambiguous

**Hence**: Multiple definitions of a “jet”
# Two classes of algorithms

<table>
<thead>
<tr>
<th>Class 1: recombination</th>
<th>Cass 2: cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successive recombinations of the “closest” ((a)) pair of particle</td>
<td>find directions of energy flow (\equiv) stable cones ((b))</td>
</tr>
<tr>
<td>Nice perturbative behaviour</td>
<td>Small sensitivity to soft radiation (UE,PU)</td>
</tr>
<tr>
<td>Often used in (e^\pm e^\pm, e^\pm p)</td>
<td>Often used in (pp)</td>
</tr>
</tbody>
</table>

\((a)\) **Distance:**

\[
k_t: \quad d_{i,j} = \min(k_{t,i}^2, k_{t,j}^2)(\Delta \phi_{i,j}^2 + \Delta y_{i,j}^2)
\]

Aachen/Cam.: \[
\quad d_{i,j} = \Delta \phi_{i,j}^2 + \Delta y_{i,j}^2
\]

\((b)\) **stable cones** (radius \(R\)) such that:

the total momentum of its contents points in the direction of its centre
How the cone works...

- Seeded (iterative) approaches: iterate from an initial position until stable
  - seed = initial particle
  - seed = midpoint between stable cones found at first step
- One has to deal with overlapping stable cones: 2 subclasses
Seeded (iterative) approaches: iterate from an initial position until stable
- seed = initial particle
- seed = midpoint between stable cones found at first step

Class 2(a): cone with split-merge (ex.: JetClu, Atlas, MidPoint):
\[
\tilde{p}_{t,\text{shared}} > f \tilde{p}_{t,\text{min}}
\]
\[
\tilde{p}_{t,\text{shared}} \leq f \tilde{p}_{t,\text{min}}
\]
Seeded (iterative) approaches: iterate from an initial position until stable
- seed = initial particle
- seed = midpoint between stable cones found at first step

Class 2(a): cone with split-merge (ex.: JetClu, Atlas, MidPoint):
\[ \tilde{p}_{t,\text{shared}} > f\tilde{p}_{t,\text{min}} \]
\[ \tilde{p}_{t,\text{shared}} \leq f\tilde{p}_{t,\text{min}} \]

Class 2(b): cone with progressive removal (ex.: Iterative Cone)
- iterate from the hardest seed
- remove the stable cone as a jet and start again

Idea: “regular/circular” jets
Progress in jet definitions

Recombination algorithms

- 20th century: the $k_t$ algorithm is too slow – $O(N^3)$
- Today: fast implementation using algorithmic geometry – $O(N \log(N))$ (the 'fast' in FastJet)

[Cacciari, Salam, 2006]

Cone algorithms

- 20th century: infrared-and/or-collinear (IRC) unsafe
- Today:
  - SISCone: IRC-safe replacement for JetClu, MidPoint-type algs.
    [Salam, 2007]
  - anti-$k_t$: IRC-safe replacement for Iterative-Cone algorithms
    [Cacciari, Salam, 2008]
SISCones and anti-$k_t$
QCD probability for gluon bremsstrahlung at angle $\theta$ and $\perp$-mom. $k_t$:

$$dP \propto \alpha_s \frac{d\theta}{\theta} \frac{dk_t}{k_t}$$

Two divergences:

- Collinear
- Soft

$\theta \approx 0$

$p_t$

$k_t \ll p_t$
QCD probability for gluon bremsstrahlung at angle $\theta$ and $\perp$-mom. $k_t$:

$$dP \propto \alpha_s \frac{d\theta}{\theta} \frac{dk_t}{k_t}$$

Two divergences:

Collinear

Soft

For pQCD to make sense, the (hard) jets (or stable cones) should not change when

- one has a collinear splitting
  i.e. replaces one parton by two at the same place $(\eta, \phi)$
- one has a soft emission i.e. adds a very soft gluon
Random “hard+soft” events:

- JetClu, ATLAS cone: 50% failure
- MidPoint, iter. cone: 15% failure

Midpoint and the iterative cone IR or Collinear unsafe† at $\mathcal{O}(\alpha_s^4)$ ($\mathcal{O}(\alpha_s^3)$ for JetClu)

<table>
<thead>
<tr>
<th>Observable</th>
<th>1st miss cones at</th>
<th>Last meaningful order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inclusive jet cross section</td>
<td>NNLO</td>
<td>NLO</td>
</tr>
<tr>
<td>3 jet cross section</td>
<td>NLO</td>
<td>LO (NLO in NLOJet)</td>
</tr>
<tr>
<td>$W/Z/H + 2$ jet cross sect.</td>
<td>NLO</td>
<td>LO (NLO in MCFM)</td>
</tr>
<tr>
<td>jet masses in 3 jets</td>
<td>LO</td>
<td>none (LO in NLOJet)</td>
</tr>
</tbody>
</table>

⇒ The IR-unsafety issue will matter at LHC

+ We do not want the theoretical efforts to be wasted
SISCone vs. MidPoint

Inclusive (midpoint/SISCOne-1)

\[ \frac{d\sigma_{\text{midpoint}}}{dp_t} / \frac{d\sigma_{\text{SISCOne}}}{dp_t} - 1 \]

- Hadron-level (with UE)
- Hadron-level (no UE)
- Parton-level

Pythia 6.4, \( R=0.7, f=0.5, |y|<0.7 \)

Masses in 3-jet events

- Mass spectrum of jet 2
- Relative difference for \( \frac{d\sigma}{dM} \)

**Inclusive cross-section:**
- Effect of a few percents
- Less sensitivity to the UE

**More exclusive processes:** effects \( \sim 45\% \) (Important for LHC!)

**HI:** mass \( \leftrightarrow \) substructure \( \leftrightarrow \) quenching
Come back to recombination-type algorithms:

\[ d_{ij} = \min(k_{t,i}^{2p}, k_{t,j}^{2p}) \left( \Delta \phi_{ij}^2 + \Delta \eta_{ij}^2 \right) \]

- \( p = 1 \): \( k_t \) algorithm
- \( p = 0 \): Aachen/Cambridge algorithm
- \( p = -1 \): anti-\( k_t \) algorithm  
  [M.Cacciari, G.Salam, G.S., JHEP 04 (08) 063]
Come back to recombination-type algorithms:

\[ d_{ij} = \min(k_{t,i}^{2p}, k_{t,j}^{2p}) \left( \Delta \phi_{ij}^2 + \Delta \eta_{ij}^2 \right) \]

- \( p = 1 \): \( k_t \) algorithm
- \( p = 0 \): Aachen/Cambridge algorithm
- \( p = -1 \): anti-\( k_t \) algorithm  
  [M.Cacciari, G.Salam, G.S., JHEP 04 (08) 063]

Why should that be related to the iterative cone ?!?

- “large \( k_t \) \( \Rightarrow \) small distance”
  - \textit{i.e.} hard partons “eat” everything up to a distance \( R \)
  - \textit{i.e.} circular/regular jets, jet borders unmodified by soft radiation
- infrared and collinear safe
Hard event + homogeneous soft background

anti-$k_t$ is soft-resilient
FastJet is an interface for running jet clustering
FastJet is an interface for running jet clustering

Jet algorithms:

- **Native**: recombination algs. (fast implementation)
  - $k_t$
  - Cambridge/Aachen
  - anti-$k_t$
  - $e^+e^-$ algorithms in preparation

Plugins:
- SISCone
- CDF JetClu and CDF MidPoint (IRC unsafe)
- PxCone (IRC unsafe)
- D0 run II cone (IRC unsafe)
FastJet is an interface for running jet clustering

Jet algorithms:

- **Native**: recombination algs. (fast implementation)
  - $k_t$
  - Cambridge/Aachen
  - anti-$k_t$
  - $e^+e^-$ algorithms in preparation

- **Plugins**:
  - SISCone
  - CDF JetClu and CDF MidPoint (IRC unsafe)
  - PxCone (IRC unsafe)
  - D0 run II cone (IRC unsafe)

- Computation of jet areas and pileup subtraction
**Idea:** use geometric arguments

<table>
<thead>
<tr>
<th></th>
<th>recomb.</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>before:</td>
<td>Naive: $O(N^3)$</td>
<td>Naive: $O(N2^N)$</td>
</tr>
<tr>
<td>now:</td>
<td></td>
<td>Midpoint: $O(N^3)$</td>
</tr>
</tbody>
</table>
**Idea**: use geometric arguments

<table>
<thead>
<tr>
<th></th>
<th>recomb.</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>before:</td>
<td>Naive: $O(N^3)$</td>
<td>Naive: $O(N2^N)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Midpoint: $O(N^3)$</td>
</tr>
<tr>
<td>now:</td>
<td>Factorisation: $O(N^2)$</td>
<td></td>
</tr>
</tbody>
</table>

One can factorise the $k_t$-dependent part
For the purely geometric part ($\equiv$ Cam/Aachen): iteration less costly

$\Rightarrow O(N^2)$
### Algorithm speed

**Idea:** use geometric arguments

<table>
<thead>
<tr>
<th></th>
<th>recomb.</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>before:</strong></td>
<td>Naive: $O(N^3)$</td>
<td>Naive: $O(N^{2N})$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Midpoint: $O(N^3)$</td>
</tr>
<tr>
<td><strong>now:</strong></td>
<td>Factorisation:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$O(N^2)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voronoi: $O(N \log(N))$</td>
<td></td>
</tr>
</tbody>
</table>

**Dynamic Nearest Neighbour using the Voronoi diagram**

![Voronoi diagram](image)

[M. Cacciari, G. Salam, 06]
Algorithm speed

**Idea:** use geometric arguments

<table>
<thead>
<tr>
<th></th>
<th>recomb.</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>before:</td>
<td>Naive: $O(N^3)$</td>
<td>Naive: $O(N2^N)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Midpoint: $O(N^3)$</td>
</tr>
<tr>
<td>now:</td>
<td>Factorisation: $O(N^2)$</td>
<td>SISCone: $O(N^2 \log(N))$</td>
</tr>
<tr>
<td></td>
<td>Voronoi: $O(N \log(N))$</td>
<td></td>
</tr>
</tbody>
</table>

Every enclosure moved to touch two points

Enumerate enclosures $\equiv$ enumerate pairs of points
SISCones (at least) as fast as Midpoint (with a 1 GeV seed threshold)

FastJet $k_t$ much faster than KtJet ($O(N^3)$)

anti-$k_t \approx k_t$ (still faster than SISCones)
Clustering done using 3 major classes

- **Class #1**: `fastjet::PseudoJet`
  - Used to deal with 4-vectors

- **Class #2**: `fastjet::JetDefinition(“algorithm”, “parameters”)`
  - Used to define the clustering recipe i.e. algorithm + parameters (e.g. $R$)

- **Class #3**: `fastjet::ClusterSequence(vector⟨fastjet::PseudoJet⟩, fastjet::JetDefinition)`
  - Really perform the clustering
Example: Cluster particles from the command line and print out jets

```cpp
#include <iostream>
#include <vector>
#include <fastjet/PseudoJet.hh>
#include <fastjet/ClusterSequence.hh>
using namespace std;

void main(){
    // read the particles
    vector<fastjet::PseudoJet> particles;
    double px, py, pz, E;
    while (cin >> px >> py >> pz >> E)
        particles.push_back(fastjet::PseudoJet(px, py, pz, E));

    // declare a jet definition
    double R = 0.5;
    fastjet::JetDefinition jet_def(kt_algorithm, R);

    // perform the clustering
    fastjet::ClusterSequence clust_seq(particles, jet_def);

    // retrieve the jets and print them out
    double ptmin = 0.0;
    vector<fastjet::PseudoJet> jets = sorted_by_pt(clust_seq.inclusive_jets(ptmin));

    for (unsigned int i=0;i<jets.size();i++)
        cout << jets[i].perp() << " " << jets[i].rap() << " " << jets[i].phi() << endl;
}
```
Example: Cluster particles from the command line and print out jets

```cpp
#include <iostream>
#include <vector>
#include <fastjet/PseudoJet.hh>
#include <fastjet/ClusterSequence.hh>
#include <fastjet/SISConePlugin.hh>

using namespace std;

void main(){
    // read the particles
    vector<fastjet::PseudoJet> particles;
    double px, py, pz, E;
    while (cin >> px >> py >> pz >> E)
        particles.push_back(fastjet::PseudoJet(px,py,pz,E));

    // declare a jet definition
    double R = 0.5;
    fastjet::JetDefinition jet_def = new SISConePlugin(R,0.75);

    // perform the clustering
    fastjet::ClusterSequence clust_seq(particles, jet_def);

    // retrieve the jets and print them out
    double ptmin = 0.0;
    vector<fastjet::PseudoJet> jets = sorted_by_pt(clust_seq.inclusive_jets(ptmin));

    for (unsigned int i=0;i<jets.size();i++)
        cout << jets[i].perp() << " " << jets[i].rap() << " " << jets[i].phi() << endl;

    delete jet_def.plugin();
}
```

Grégory Soyez  
Yale, USA, July 1st 2008
Jet area

Everyone has an idea of what a jet area is
but can we define that properly?

[M. Cacciari, G. Salam, G.S., JHEP 04 (2008) 5]
**Idea:** add soft particles (ghosts) and look in which jets they are caught.

**jet area = region where it catches ghosts**
**Idea:** add soft particles (ghosts) and look in which jets they are caught

Jet area = region where it catches ghosts

2 definitions

- **Passive area**
  add one ghost and look where it ends. repeat to cover the \((y, \phi)\) plane

- **Active area**
  add a large amount of ghosts and cluster everything
  also gives purely ghosted jets
**Area definition**

- **Idea**: add soft particles (ghosts) and look in which jets they are caught

  jet area = region where it catches ghosts

- 2 definitions
  - **Passive area**
    - add one ghost and look where it ends. repeat to cover the \((y, \phi)\) plane
  - **Active area**
    - add a large amount of ghosts and cluster everything
      - also gives purely ghosted jets

- Both definitions agree for dense events
- Both *practical* and *tractable* analytically
Hard event + ghost added at each point of the grid
Jet areas = shaded regions

\( k_t \) algorithm, passive area

\( k_t \) algorithm, active area
**Examples: 1-particle cases**

<table>
<thead>
<tr>
<th></th>
<th>$k_t$</th>
<th>Aac/Cam</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passive</td>
<td><img src="image1" alt="Passive kt" /></td>
<td><img src="image2" alt="Passive Aac/Cam" /></td>
<td><img src="image3" alt="Passive cone" /></td>
</tr>
<tr>
<td></td>
<td>$\pi R^2$</td>
<td>$\pi R^2$</td>
<td>$\pi R^2$</td>
</tr>
<tr>
<td>Active</td>
<td><img src="image4" alt="Active kt" /></td>
<td><img src="image5" alt="Active Aac/Cam" /></td>
<td><img src="image6" alt="Active cone" /></td>
</tr>
</tbody>
</table>
### Examples: 1-particle cases

<table>
<thead>
<tr>
<th>Passive</th>
<th>$k_t$</th>
<th>Aac/Cam</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Circle" /></td>
<td>$\pi R^2$</td>
<td>$\pi R^2$</td>
<td>$\pi R^2$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Active</th>
<th>$\frac{A_{\text{hard}}}{\pi R^2}$</th>
<th>$\frac{A_{\text{ghost}}}{\pi R^2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image2" alt="Graph" /></td>
<td>$0.812 \pm 0.277$</td>
<td>$0.554 \pm 0.174$</td>
</tr>
<tr>
<td><img src="image3" alt="Graph" /></td>
<td>$0.814 \pm 0.261$</td>
<td>$0.551 \pm 0.176$</td>
</tr>
</tbody>
</table>

\[
\frac{A_{\text{hard}}}{\pi R^2} = 0.25
\]
### Examples: 1-particle cases

<table>
<thead>
<tr>
<th></th>
<th>$k_t$</th>
<th>Aac/Cam</th>
<th>cone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passive</td>
<td></td>
<td>$\pi R^2$</td>
<td>$\pi R^2$</td>
</tr>
<tr>
<td>Active</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\frac{A_{\text{hard}}}{\pi R^2} \approx 0.812 \pm 0.277$</td>
<td>$\frac{A_{\text{hard}}}{\pi R^2} \approx 0.814 \pm 0.$</td>
<td>$\frac{A_{\text{ghost}}}{\pi R^2} \approx 0.554 \pm 0.174$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\frac{A_{\text{ghost}}}{\pi R^2} \approx 0.551 \pm 0.$</td>
<td></td>
</tr>
</tbody>
</table>

$\mathcal{A}_{\text{ghost}}$ depends on $f$

possible monster jets!
2-particle cases

Passive area: 1 hard particle + 1 soft

\[ 0 < \Delta_{12} < R/2 \]
\[ R/2 < \Delta_{12} < R \]
\[ R < \Delta_{12} < 2R \]
**Active area**: 1 hard particle + 1 soft: analytic result for cone only

Alltogether, we have:
- Area $\neq$ cst. $\pi R^2$
- $\Delta_{12}$ dependence under control
QCD probability of emitting a small-angle soft gluon:

$$\frac{dP}{d\Delta_{12} dp_{t,2}} = C_{F,A} \frac{2\alpha_s}{\pi} \frac{1}{\Delta_{12}} \frac{1}{p_{t,2}}$$

Hence the average area is

$$\langle A(p_{t,1}, R) \rangle = A_{1\text{hard}}(R) + \int d\Delta dp_{t,2} \frac{dP}{d\Delta_{12} dp_{t,2}} [A_{\text{hard+1 soft}}(\Delta, R) - \pi R^2]$$
QCD probability of emitting a small-angle soft gluon:

\[ \frac{dP}{d\Delta_{12} dp_{t,2}} = C_{F,A} \frac{2\alpha_s}{\pi} \frac{1}{\Delta_{12}} \frac{1}{p_{t,2}} \]

Hence the average area is

\[
\langle A(p_{t,1}, R) \rangle = A_{1\text{hard}}(R) + \int d\Delta d p_{t,2} \left( \frac{dP}{d\Delta_{12} dp_{t,2}} \left[ A_{\text{hard+1 soft}}(\Delta, R) - \pi R^2 \right] \right)
\]

\[
= C_{F,A} \frac{\pi b_0}{\pi} \log \left( \frac{\alpha_s(Q_0)}{\alpha_s(Rp_t)} \right) \pi R^2 d
\]

- Scaling violation
Area scaling violations

QCD probability of emitting a small-angle soft gluon:

\[ \frac{dP}{d\Delta_{12} dp_{t,2}} = C_{F,A} \frac{2\alpha_s}{\pi} \frac{1}{\Delta_{12}} \frac{1}{p_{t,2}} \]

Hence the average area is

\[ \langle A(p_{t,1}, R) \rangle = A_{1\text{hard}}(R) + \int d\Delta d p_{t,2} \frac{dP}{d\Delta_{12} dp_{t,2}} \left[ A_{\text{hard+1 soft}}(\Delta, R) - \pi R^2 \right] \]

\[ = \frac{C_{F,A}}{\pi b_0} \log \left( \frac{\alpha_s(Q_0)}{\alpha_s(Rp_t)} \right) \pi R^2 d \]

- Scaling violation
- gluon > quark
Area scaling violations

QCD probability of emitting a small-angle soft gluon:

\[
\frac{dP}{d\Delta_{12}dp_{t,2}} = C_{F,A} \frac{2\alpha_s}{\pi} \frac{1}{\Delta_{12}} \frac{1}{p_{t,2}}
\]

Hence the average area is

\[
\langle A(p_{t,1}, R) \rangle = A_{\text{hard}}(R) + \int d\Delta dp_{t,2} \frac{dP}{d\Delta_{12}dp_{t,2}} \left[ A_{\text{hard}+1 \text{ soft}}(\Delta, R) - \pi R^2 \right]
\]

\[
= \frac{C_{F,A}}{\pi b_0} \log \left( \frac{\alpha_s(Q_0)}{\alpha_s(Rp_t)} \right) \pi R^2 d
\]

<table>
<thead>
<tr>
<th></th>
<th>passive</th>
<th>active</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k_t )</td>
<td>0.5638</td>
<td>0.519</td>
</tr>
<tr>
<td>Cam</td>
<td>0.07918</td>
<td>0.0865</td>
</tr>
<tr>
<td>SISCone</td>
<td>-0.06378</td>
<td>0.1246</td>
</tr>
<tr>
<td>( \text{anti-}k_t )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- Scaling violation
- gluon > quark
- with known LO anomalous dimension
"Real-life" anomalous dimension

Herwig simulations:
at hadron+UE level:
area vs. $p_t$ of the jet

- good agreement
  with LO predictions

- for flucrs. too

- $k_t$ bigger
  $\Rightarrow$ NLO?
Area histograms

(a) Pythia 6.4
$P_{t,\text{min}}=1$ TeV
2 hardest jets
$|y|<2$, $R=1$

(b) $k_t$
Cam/Aachen
SIScone
anti-$k_t$ (/5)

(c) $k_t$

(d) $k_t$

(e) $k_t$

(f) $k_t$
What can area be used for?

Dense event with pile-up (or uniform soft background):

$k_t$ algorithm, $R=0.5$
For pure pile-up jets: Area $\propto p_t$ of the jet

$p_t$/area is constant $\rightarrow \rho = \text{median } p_t$/area
What can area be used for?

Dense event with pile-up (or uniform soft background):

- For pure pile-up jets: Area $\propto p_t$ of the jet
- $p_t$/area is constant $\rightarrow \rho = \text{median } p_t$/area

**Area can be used to subtract pileup:**

$$p_{t,\text{corrected}} = p_{t,\text{bare}} - \rho \text{ area}$$
Implementation in FastJet using the `AreaDefinition` and `sf ClusterSequenceArea` classes

```cpp
// declare usual FastJet tools: particles and jet definition
vector<fastjet::PseudoJet> particles;
fastjet::JetDefinition jet_def(kt_algorithm, R);

// define the type of area you want
fastjet::GhostedAreaSpec area_spec(maxrap_ghost, num_repeat, ghost area);
fastjet::AreaDefinition area_def(active_area, area_spec);

// perform the clustering with area computation
fastjet::ClusterSequenceArea clust_seq(particles, jet_def, area_def);

// get the median background per unit area (i.e. rho)
double rho = clust_seq.median_pt_per_unit_area_4vector(maxrap_ghost);

// retrieve the jets and do the subtraction
vector<fastjet::PseudoJet> jets = sorted_by_pt(clust_seq.inclusive_jets(ptmin));
fastjet::PseudoJet jet_sub = jets[0] - rho * clust_seq.area_4vector(jets[0]);
```
Subtraction in action

$\bar{t}t + W$

$\bar{t}t \rightarrow \ell^+ \nu \ell b + q \bar{q} b$

$(W \rightarrow q\bar{q})$

$k_t, R=0.4$

no pileup

reconstructed $W$ / top mass [GeV]

Cam/Aachen, $R=0.4$

reconstructed $W$ / top mass [GeV]

SISCone, $R=0.4, f=0.5$

no pileup  $\Rightarrow$ good result

LHC at high lumi

Grégory Soyez

Yale, USA, July 1st 2008

FastJet – p. 30/33
Subtraction in action

$t\bar{t} + W$  

$(t\bar{t} \rightarrow \ell^+ \nu \ell b + q\bar{q}b)$  

$(W \rightarrow q\bar{q})$

\[ W \]

\[ \text{Cam/Aachen, } R=0.4 \]

LHC at high lumi

no pileup  ⇒  good result

⇒  no subtraction effect

Grégory Soyez  
Yale, USA, July 1st 2008  
FastJet – p. 30/33
Subtraction in action

\( \bar{t}t + W \) 
\( (\bar{t}t \rightarrow \ell^+ \nu \ell b + q\bar{q}b) \)

\( W \rightarrow q\bar{q} \)

LHC at high lumi

no pileup \( \Rightarrow \) good result

no pileup, sub \( \Rightarrow \) no subtraction effect

pileup \( \Rightarrow \) poor result
**Subtraction in action**

\[ t\bar{t} + W \quad (t\bar{t} \rightarrow \ell^+ \nu\ell b + q\bar{q}b) \]

\[ W \rightarrow q\bar{q} \]

*Cam/Aachen, R=0.4, pileup, sub*

\[ k_t, R=0.4 \]

no pileup, sub

\[ \text{no pileup} \]

\[ \text{reconstructed W / top mass [GeV]} \]

\[ \frac{1}{N} \frac{dN}{dm} \text{ [GeV}^{-1}] \]

\[ \text{reconstructed W / top mass [GeV]} \]

\[ \text{reconstructed W / top mass [GeV]} \]

\[ \text{reconstructed W / top mass [GeV]} \]

**LHC at high lumi**

- no pileup \( \Rightarrow \) good result
- \( \Rightarrow \) no subtraction effect
- pileup \( \Rightarrow \) poor result
- \( \Rightarrow \) subtraction works

Grégory Soyez

Yale, USA, July 1st 2008

FastJet – p. 30/33
For heavy-ion collisions at LHC:

- Large background ($\sim 250$ GeV/unit area)
- Not really uniform
For heavy-ion collisions at LHC:

- Large background ($\sim 250$ GeV/unit area)
- Not really uniform
- Background: $p_t/\text{area} \approx \text{parabolic in rapidity}$

![Graph showing $p_t/\text{area}$ as a function of rapidity](image-url)
For heavy-ion collisions at LHC:

- Large background ($\sim 250$ GeV/unit area)
- Not really uniform
- Background: $p_t/\text{area} \approx \text{parabolic in rapidity}$
- After subtraction:

\[
\begin{align*}
\text{LHC, Pb Pb, } \sqrt{s} &= 5.5 \text{ TeV} \\
\text{Hydjet, } dN_{\text{ch}}/dy &= 1600
\end{align*}
\]
Back reaction: when adding soft background to a hard event some hard parton can be associated to another hard jet.

- **Point-like** or **diffuse** background
- **gain**: $p_2$ gained when adding $p_1$

![Diagram showing point-like or diffuse background gain]

- **loss**: $p_2$ lost when adding $p_1$
Under **analytic control!**

- **Probability of gain/loss** ($\rho \equiv$ background density)

\[
dP^{(G,L)} = \frac{dP^{(G,L)}}{dp_{t,2}} = \frac{2\alpha_s C_{F,A}}{\pi} \frac{1}{p_{t,2}} b^{(G,L)}(p_{t,2}/\rho)
\]

\[p_{t,m} \ll p_{t,2} \ll p_{t,1} : \begin{cases} \propto p_{t,m}/p_{t,2} \quad & \text{for } k_t, \text{Cam} \\ \approx 0 \quad & \text{for } \text{anti–}k_t, \text{SIS Cone} \end{cases}\]
Back-reaction (2)

Under analytic control!

- Probability of gain/loss ($\rho \equiv$ background density)

\[
\frac{dP^{(G,L)}}{dp_{t,2}} = \frac{2\alpha_s C_{F,A}}{\pi} \frac{1}{p_{t,2}} b^{(G,L)}(p_{t,2}/\rho)
\]

- Shift in $p_t$ due to back-reaction: $\Delta p_t^{(G−L)}$

<table>
<thead>
<tr>
<th>$\Delta p_t^{(B)}$ (GeV)</th>
<th>1/N dN/d$p_t$ (GeV$^{-1}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R=1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>0.001</td>
</tr>
</tbody>
</table>

anti-$k_t \ll$ SIS Cone $< k_t$, Cam
Probability of gain/loss ($\rho \equiv$ background density)

$$\frac{dP^{(G,L)}}{dp_{t,2}} = \frac{2\alpha_s C_{F,A}}{\pi} \frac{1}{p_{t,2}} b^{(G,L)}(p_{t,2}/\rho)$$

Shift in $p_t$ due to back-reaction: $\Delta p_{t}^{(G-L)}$

$$\langle \Delta p_{t}^{(G-L)} \rangle = B \cdot \rho \cdot \frac{C_{F,A}}{\pi b_0} \cdot \log \left( \frac{\alpha_s(\rho R^3)}{\alpha_s(p_{t,1} R)} \right)$$

<table>
<thead>
<tr>
<th>$B/\pi R^2$</th>
<th>pointike</th>
<th>diffuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_t$</td>
<td>$\sqrt{3}/4 \approx 0.14$</td>
<td>$\approx 0.10$</td>
</tr>
<tr>
<td>Cam</td>
<td>$\sqrt{3}/4 \approx 0.14$</td>
<td>$\approx 0.10$</td>
</tr>
<tr>
<td>SIS Cone</td>
<td>0 (+NLO)</td>
<td>0 (+NLO)</td>
</tr>
<tr>
<td>anti-$k_t$</td>
<td>0 (+power corr.)</td>
<td>0 (+power corr.)</td>
</tr>
</tbody>
</table>