

SpartyJet

Generic routines for Jet analysis



(this is Sparty!)

Pierre-Antoine Delsart
Université de Montréal

Kurtis L. Geerlings & Joey Huston
Michigan State University

Introduction

Goal :

Provide a set of routine for both Jets users and developers

Fast to run (no need of heavy framework like Athena)

Easy to use (driven through ROOT scripts)

Flexible (adapt to any input – allow in-depth analysis)

Use cases

SpartyJet is already used to :

- Analyze and compare lots of jet algorithm
compare inter-experiments jets CMS/CDF/ATLAS
compare with theorists' algorithms
- Develop and test new algorithms
- Develop physics analysis
(see Alessandro's talk)

Implementation summary

Tried to write it “modulable”
Inspired by Atlas Jet software

3 Basic parts

Input Classes

A simple interface to read any kind of input into's SpartyJet format

Jet Sequence classes

Each jet algo is a sequence of tools (preselection, jet finding, moment calculation, etc...)

Output classes

ROOT Ntuple

Plus

Basic simple objects (Jet, JetMoment, ...)

A wrapper class '**JetBuilder**' to hold everything together

Main Classes

InputMaker

```
fillInput (int eventn,  
           Jet::jet_list_t &inputList)
```

Reads a input collection of 4-vectors
initial jets list

Example implementation :

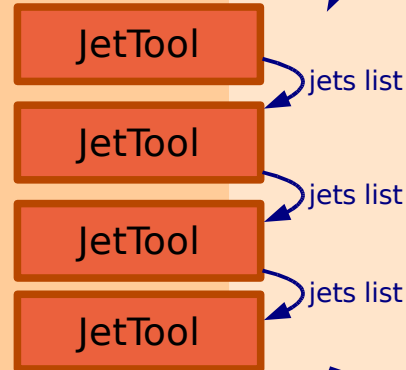
NtupleInputMaker

TextInputMaker

JetAlgorithm

```
addTool (JetTool * tool);  
execute (Jet::jet_list_t  
&inputJets,  
         Jet::jet_list_t &outputJets);
```

passes **jets list**
through a list of JetTools



JetTool

a base class.

```
execute (Jet::jet_list_t &inputJets)
```

modifies the input jet list

Example sequence :

JetSelectorTool

MinBiasInserterTool

JetKtFinderTool

JetSelectorTool

NtupleMaker

```
addJetVar (std::string jetname);  
set_data (std::string jetname,  
          Jet::jet_list_t &theJets);
```

Handle ntuple creation for arbitrary
number of jet collection identified by names.

The only place where jet algs
are actually implemented.
Simply inherit from JetTool and plug
implementation in execute()

What's available now

List of Algorithms :

CDF algorithms

JetClu

MidPoint

ATLAS standard algorithms:

Cone

Kt (fast version)

FastJet algorithms (from G. Salam, and M. Cacciari)

FastKt

Seedless Cone (SISCone)

Pythia's CellJet

+ all variants,
fully parameterizable

Also Available :

Jet Moment framework : jet Areas

Jets constituents

Timing

Example of use

- All code is **compiled** into libraries loadable into ROOT
- Easiest way of running : through ROOT scripts

1st step : configure input

```
// This is the main class
JetBuilder builder;

// configure an interface to the tree -----
//atlas::CBNTInput input;
NtupleInputMaker input(NtupleInputMaker::EtaPhiPt_vector_float);
// give variable names
input.set_prefix("cl_");
input.set_suffix("_caltopo");
input.set_variables("eta","phi","pt","e");
input.set_n_name("nc");
input.set_masslessMode(true); // consider clusters as massless
// Give a name to input :
input.set_name("InputJet");
// give input file and tree names
input.setFileTree("data/AtlasClustersJ5.root", "CollectionTree");

// Finally assign input class to the builder
builder.configure_input((InputMaker*)&input);
// -----
```

Instantiate a builder class

Configure an input class (set variable names, TTree and file name)

Associate input class to JetBuilder

Example of use II

2nd step : schedule jet algorithms

```
// Schedule 2 algorithms -----  
  
// instanciate a jet finder tool, configure it and add it to the builder  
atlas::FastKtTool * fastkt = new atlas::FastKtTool("FastKt");  
fastkt->simple_config("Standard",0.7) ;  
builder.add_default_alg(fastkt);  
  
atlas::ConeFinderTool * coneF = new atlas::ConeFinderTool("ConeFinder");  
coneF->set_config(0.7,2*GeV,0.5);  
builder.add_default_alg(coneF);  
// -----
```

3rd step : Run

```
// Other settings -----  
// configure min Pt cuts for input and output jets  
builder.set_default_cut(0,2*GeV);  
  
// Give final file and tree name  
builder.configure_output("myTree", "out.root");  
// -----  
// Run !! -----  
builder.process_events(10);  
// -----
```

Working with SpartyJet

SpartyJet comes with

- ROOT scripts examples
- PyROOT scripts example
- Analysis/visualization scripts
- **Skeleton class for adding one's own jet algorithm**

For Atlas users

- An EventView jobOption is available to extract TopoClusters and truth particles from AOD
- Works with CSC AOD's

Documentation, downloads:

www.pa.msu.edu/~huston/SpartyJet/SpartyJet.html

(very basic for now, documentation will follow)

Illustrations

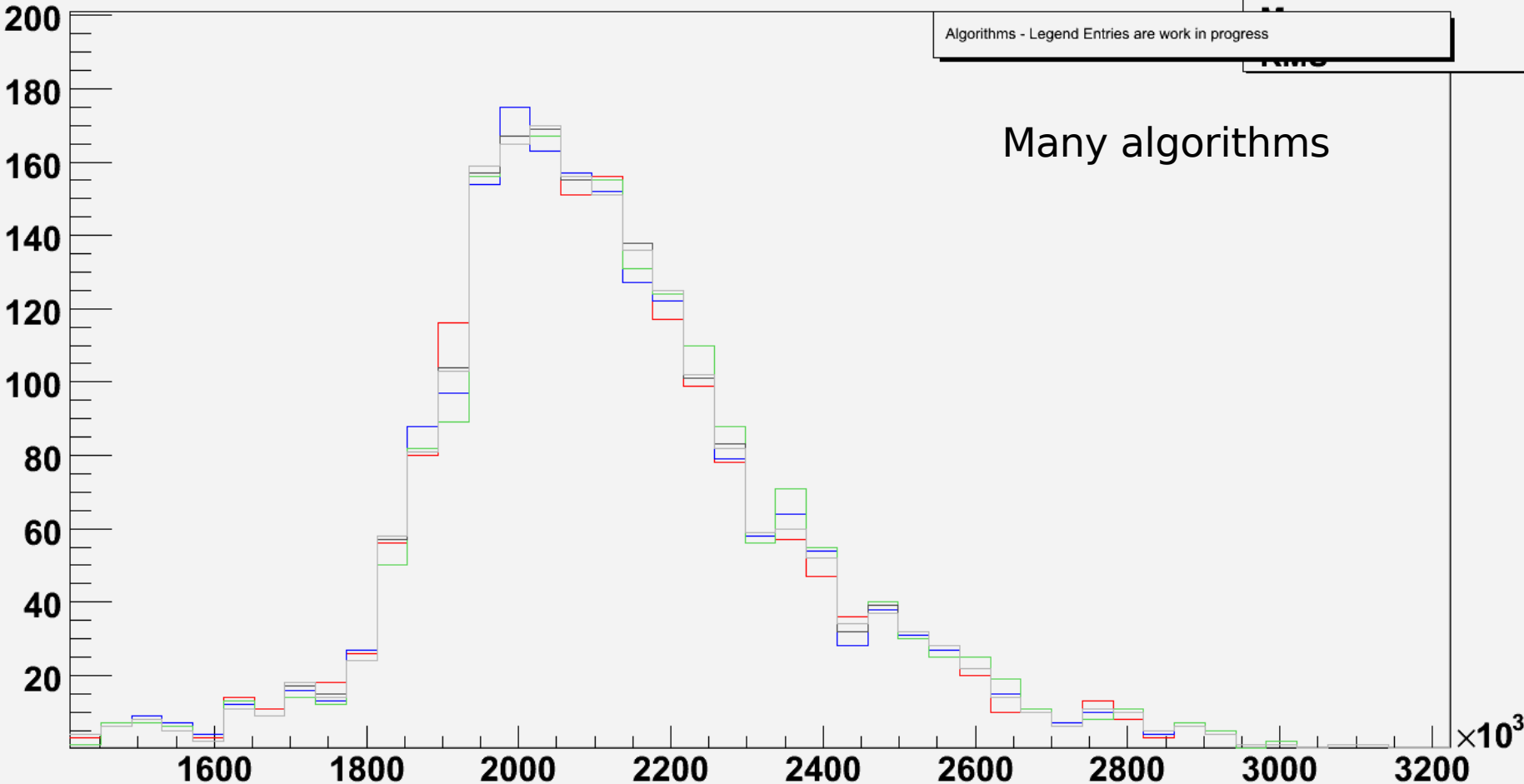
Sample used : J5 Dijets events

Pt of each jet

Plot	
Entries	0
M	0
RMS	0

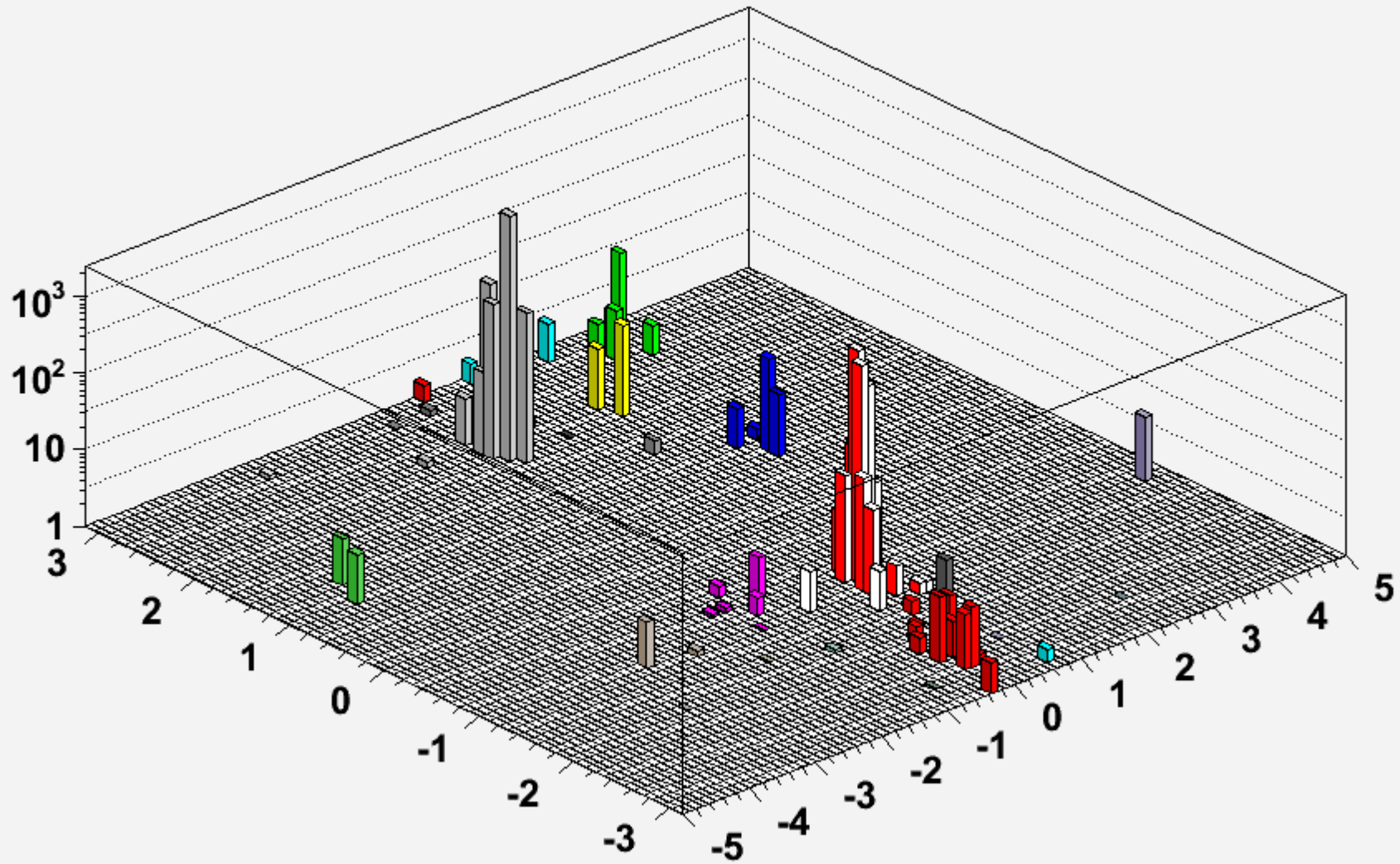
Algorithms - Legend Entries are work in progress

Many algorithms



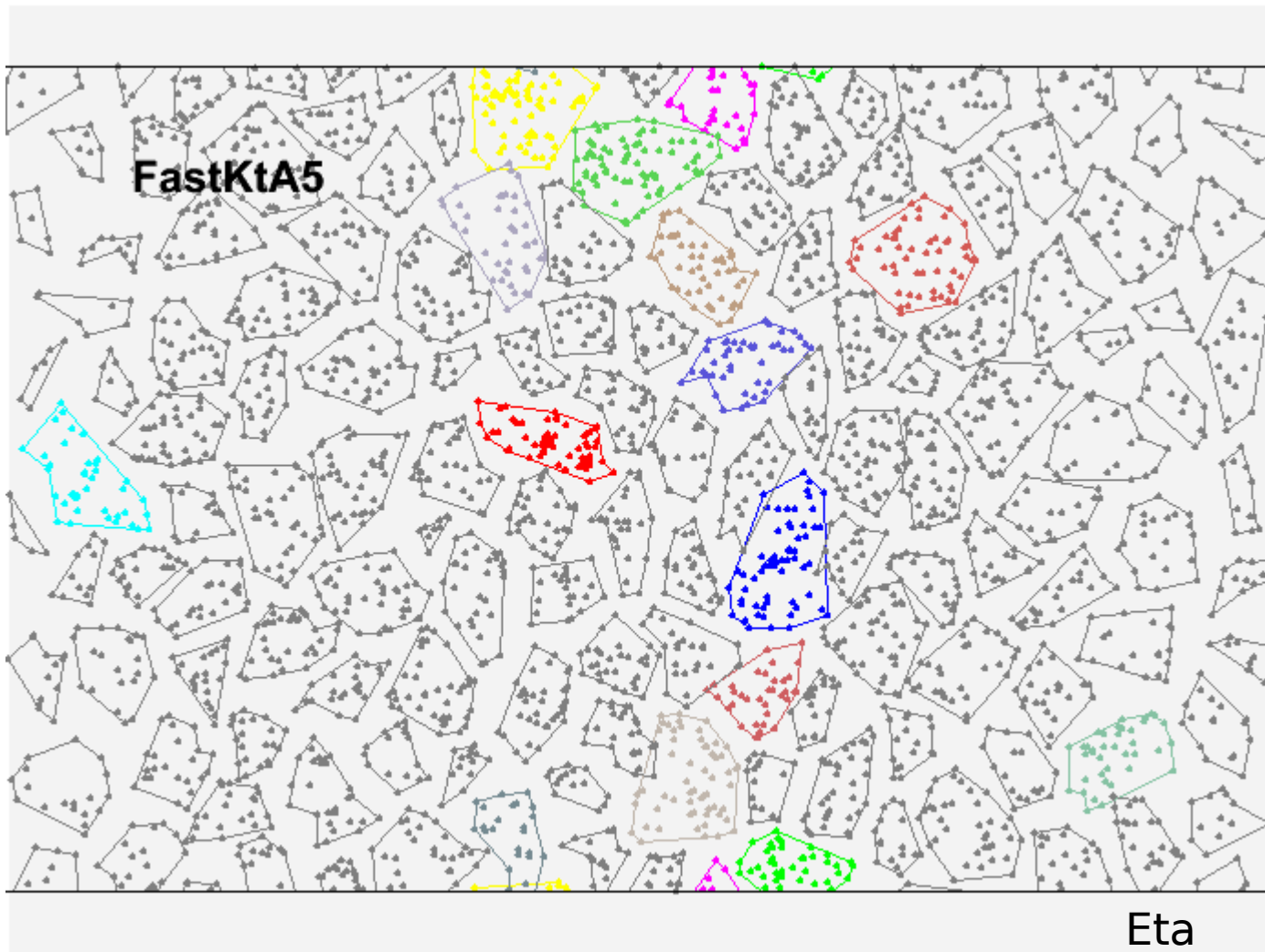
Event visualization Jets positions

Lego Plot



Constituents and jets positions
J5 sample with min-bias

Phi



Snowmass Jet Potential

SnowMass Potential

smplot	
Entries	25200
Mean x	1.38
Mean y	0.2781
RMS x	2.515
RMS y	1.752

