

SpartyJet



Kurtis Geerlings
Michigan State University

Pierre-Antoine Delsart
Université de Montréal
Joey Huston
Michigan State University

<http://www.pa.msu.edu/~huston/SpartyJet/SpartyJet.html>

What is SpartyJet?

- “a framework intended to allow for the easy use of multiple jet algorithms in collider analyses”
 - **Fast** to run, no need for heavy framework
 - **Easy** to use, basic operation is very simple
 - **Flexible**
 - ROOT-script or standalone execution
 - many different input types
 - different algorithms
 - output format

Possible Uses

- analyze and compare different algorithms
 - or the same algorithm with different parameters
- compare jets from CMS/CDF/ATLAS
- develop and test new algorithms
- Kevin Lannon and Jahred Adelman have SpartyJet working on Top Ntuples
- Andrea Messina used an older version of SpartyJet on Top Ntuples (see talk to top group from February 22, 2007)
- physics analysis

Possible Uses

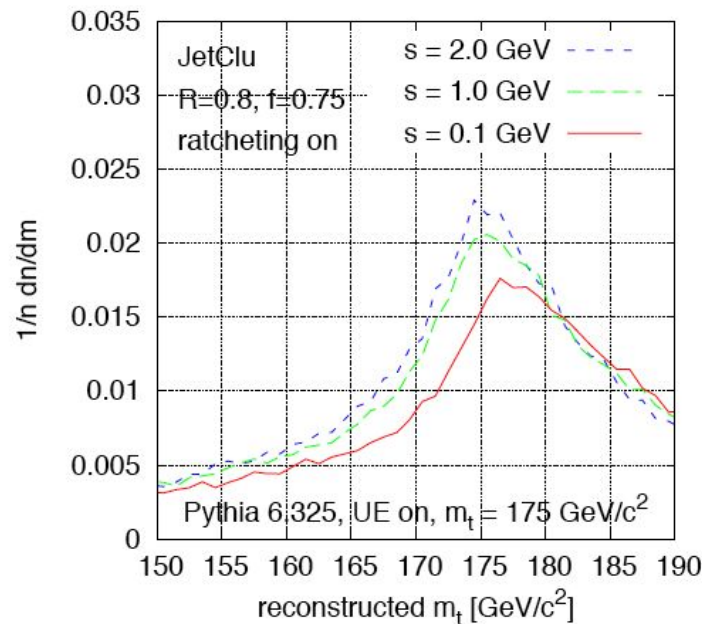
SpartyJet can be used for testing the choice of different jet algorithms for generator or detector level studies.

See for example Gavin's talk at the CDF collaboration meeting.

NLO, Jets, etc. (G. Salam, LPTHE) (p. 23)

└ Jet algorithms
└ IRC safety

JetClu's seed threshold dependence



JetClu & MidPoint use a seed threshold (s).

Seeds should just be a trick to speed up jet-finding, with no effect on physics.

IRC unsafety \rightarrow physical effect

E.g. top mass peak: shifts by 3 GeV for $0.1 < s < 2$ GeV.

Or height by 25% for $R = 0.8$

Accounted for in simulations: **but to what extent to you trust Pythia (e.g. UE) and detector details at 1 GeV level?**

e.g. event-by-event correlations between soft particles and jets

Algorithms now Available

- CDF
 - Jet Clustering
 - MidPoint (with optional second pass)
- ATLAS
 - Cone
 - Kt (fast version)
- FastJet (from Gavin Salam and Matteo Cacciari)
 - FastKt
 - Seedless Infrared Safe Cone (SISCone)
- Pythia's CellJet

all algorithms are
fully parameterizable

Types of Input

- Ntuples
 - ATLAS CBNT
 - Top Ntuples
- Text (ASCII)
 - simple list of 4-vectors
 - HepMC
- StdHep (Work in Progress, nearly finished)
- Any other Ntuple or Text Input that one writes a class for

Main SpartyJet Classes

- Input Classes
 - A simple interface to read any kind of input into SpartyJet's classes and containers
- Jet Algorithm Classes
 - Comprised of a list of Jet Tools
 - Jet Tools
 - selection cuts on input particles
 - jet finders
 - moment calculation
 - selection cuts on output jets
- Output Classes
 - ROOT Ntuple

JetBuilder

- basically a frontend to handle most of the details of running SpartyJet
- not necessary, but makes running SpartyJet **much** simpler
- Allows options that are not otherwise accessible
 - text output
 - add minimum bias events

```
gSystem->Load("libTree.so");
gSystem->Load("libs/libJetCore.so");
gSystem->Load("libs/libCDFJet.so");

StdTextInput textinput("data/J1_Clusters.dat");

JetBuilder builder;
builder.configure_input((InputMaker*)&textinput);

builder.add_default_alg( new cdf::JetClustFinder("myJetClu"));
builder.set_default_cut(0,1*GeV);

builder.configure_output("SpartyJet_Tree","data/output/simple.root");
builder.process_events(10);
```

with JetBuilder

```
TFile f("/home/delsart/SpartyJet_vwithSISCone/example/data/small.root");
TTree * tree = (TTree*) f.Get("CollectionTree");

atlas::CBNTInput input;
input.init(tree);

JetAlgorithm * alg = new JetAlgorithm("MidPointJets");

JetPtSelectorTool *selec = new JetPtSelectorTool(1*GeV);
MidPoint * midpoint = new MidPoint("TOTO");

alg->addTool((JetTool*)midpoint);
alg->addTool((JetTool*)selec);

alg->init();

NtupleMaker ntp;
ntp.addJetVar("MidPointJets");
ntp.init("JetTree","out.root");

Jet::jet_list_t injets;
Jet::jet_list_t outjets;

input->fillInput(2,injets);
alg->execute(injets, outjets);

ntp.set_data("MidPointJets", outjets);
ntp.fillJets();

clear_jetlist(injets);
clear_jetlist(outjets);

input->fillInput(5,injets);
alg->execute(injets, outjets);

ntp.set_data("MidPointJets", outjets);
ntp.fillJets();

ntp.finalize();
```

without JetBuilder

JetBuilder Example

```
gSystem->Load("libTree.so") ;  
gSystem->Load("libs/libJetCore.so");  
gSystem->Load("libs/libCDFJet.so");  
  
StdTextInput textinput("data/J1_Clusters.dat");  
  
JetBuilder builder;  
builder.configure_input((InputMaker*)&textinput);  
  
builder.add_default_alg( new cdf::JetClustFinder("myJetClu"),true);  
  
builder.set_default_cut(0,1*GeV);  
  
builder.add_text_output("data/output/text_simple.dat");  
  
builder.configure_output("SpartyJet_Tree","data/output/simple.root");  
builder.process_events(10);
```

Load the libraries that are needed

Create an input object and pass it the data file

Create a JetBuilder object and pass it the input object

Add any desired Algorithms

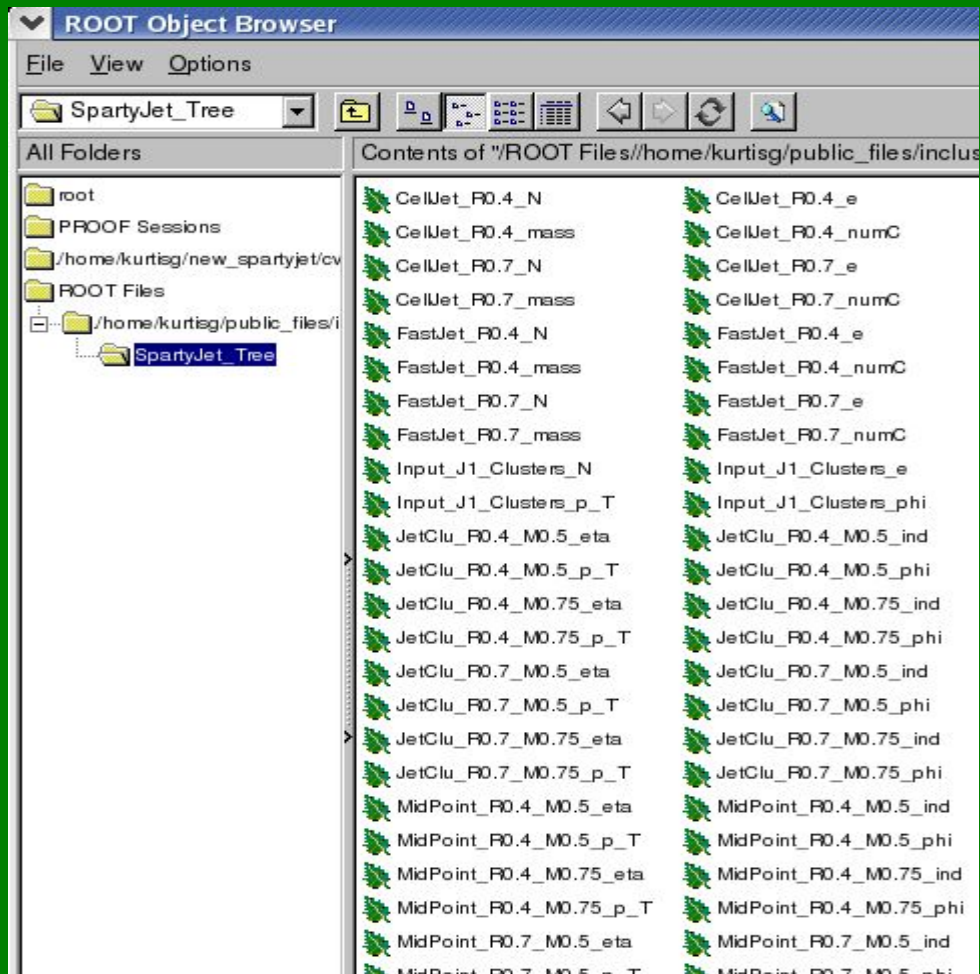
other settings, input/output cuts

Option : add text output file

Declare the output ROOT file you want

Run the JetBuilder

Root TTree Output



- branches for each algorithm
 - “name”_N
 - “name”_e
 - “name”_phi
 - “name”_eta
 - “name”_p_T
 - “name”_mass
- constituents
 - “name”_numC
 - “name”_ind
- pass information for midpoint
 - “name”_pass_num
- any other jet moments you add

Text Output

```
*****
*****
**** SpartJet Text Output File ****
****-----****
****-----****
****          Current Job Description ****
****-----****
**** Input File Info: ****
**** Object Name: calchepinput ****
****-----****
**** Algorithms: ****
**** myJetClu ****
****-----****
**** Orders: ****
**** Process 10 events, starting at event 0 ****
**** input jet Pt cut :0 GeV ****
**** output jet Pt cut :1 GeV ****
*****
*****
```

```
*****
Event 0
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
100.003 0.717967 -1.05079 2 22.0518
100.003 3.10633 2.0908 1 -8.25906e-06
*****
```

```
*****
Event 1
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
113.688 -0.336864 1.96179 1 9.57392e-07
61.1257 1.75774 -1.23997 1 -1.82939e-06
52.8011 -0.416591 -1.11013 1 5.20688e-07
*****
```

```
*****
Event 2
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
108.558 2.21862 1.77451 2 22.0539
108.558 0.526507 -1.36708 1 -1.31062e-06
*****
```

```
*****
Event 3
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
122.838 -0.471314 -3.10252 1 -1.912e-06
110.026 1.08789 0.0601439 1 -1.51729e-06
13.044 -0.962247 -0.139572 1 7.31903e-08
*****
```

```
*****
Event 4
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
209.643 0.974393 0.959828 1 -4.00107e-06
125.992 -0.936905 -2.08334 1 -1.83424e-06
85.1652 0.218748 -2.32765 1 -4.14027e-07
*****
```

```
*****
Event 5
3 four vectors
*****
myJetClu Jets
Pt eta phi n mass
105.99 -3.09708 1.93681 1 -1.50525e-05
58.3553 2.48825 -1.4819 1 1.86972e-06
52.3544 -1.19784 -0.894908 1 1.08931e-06
*****
```

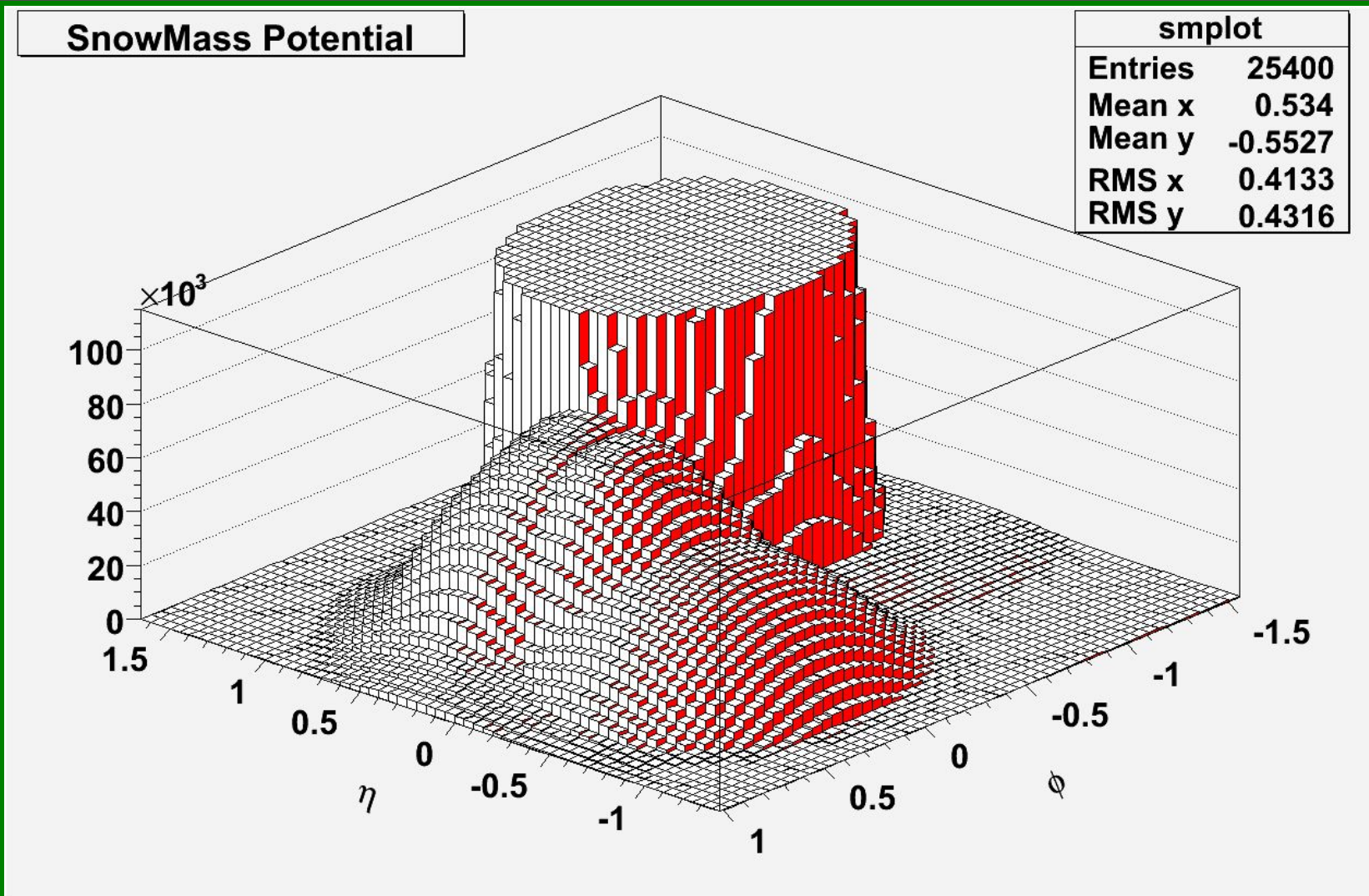
SpartyJet also contains...

- CLHEP Vector classes
- several ROOT script examples
- sample data files for all data types except StdHep
- skeleton class for adding one's own jet algorithm
- ROOT analysis script to allow for easy creation of many plots and graphs
- basic documentation describing all of the features

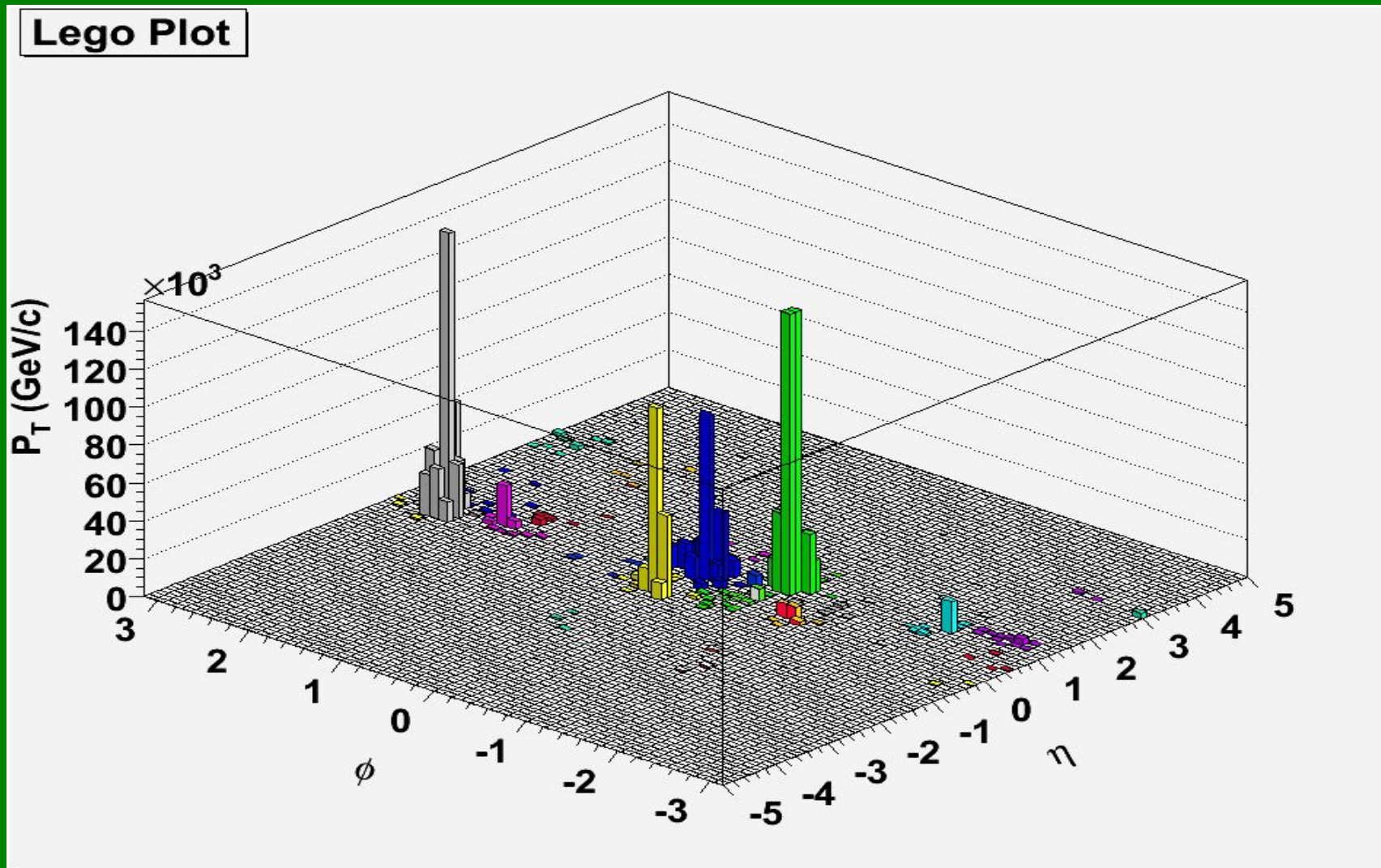
Analysis script

- SpartyJet/scripts/analysis.c
- script allows for easy access to
 - simple jet lists
 - Lego Plots
 - Snowmass potential plots
 - Z vs D split/merge plot
 - other inclusive plots (Pt,Et,mass,eta,phi...)
- this script can be compiled to increase speed

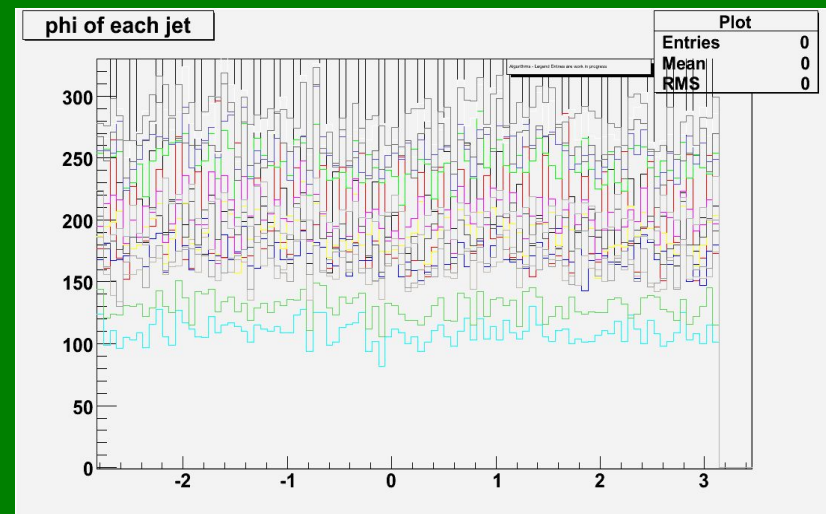
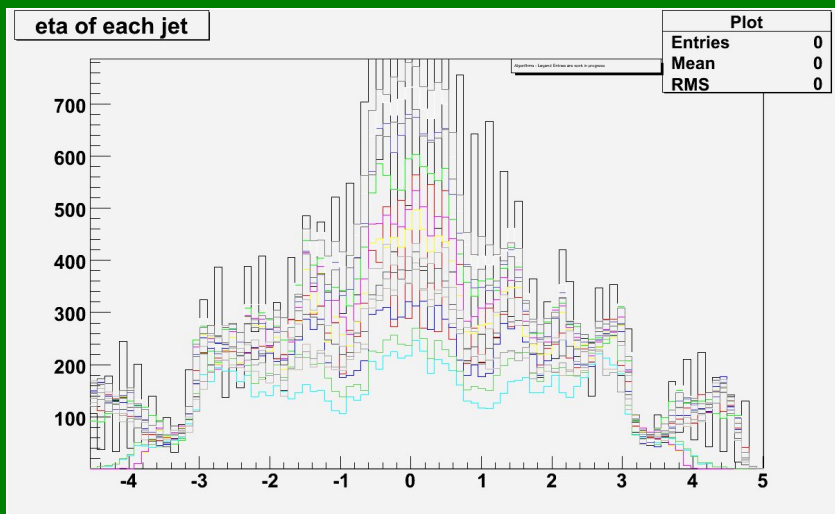
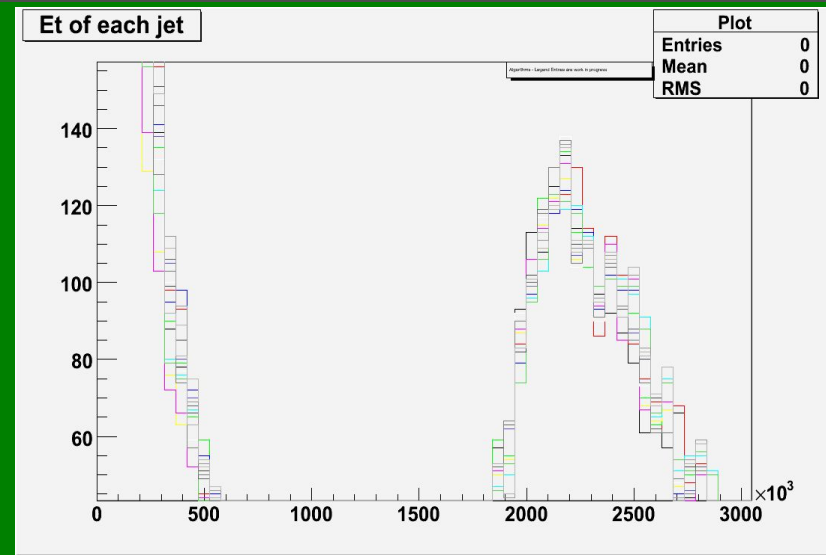
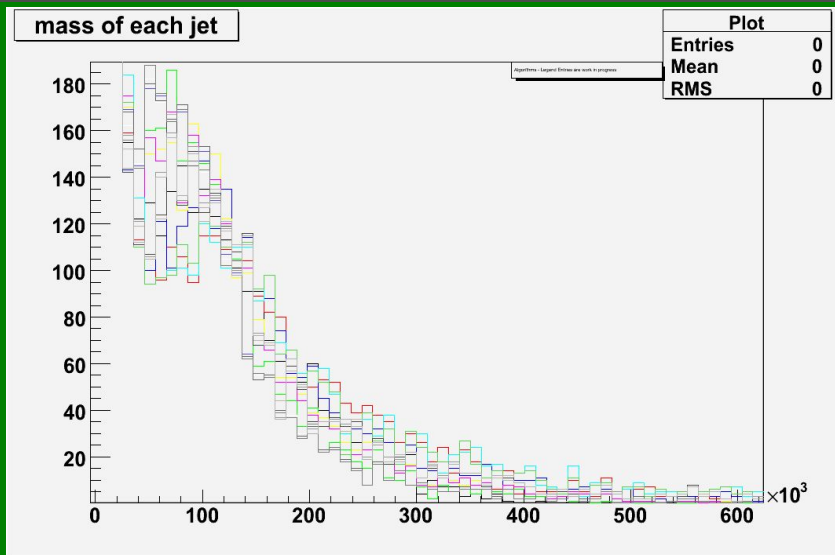
Snowmass Potential



Lego Plot



Other Plots



Simple Jet List

Et	Pt	eta	phi	mass	n
Event : 2					
CellJet_R0.7					
3.29933e+06	2.63051e+06	-0.697758	2.06004	116343	0
2.90489e+06	2.57596e+06	0.497591	-1.0774	140313	0
98907.6	76442.5	-0.716714	-1.66755	19674.9	0
68920.9	26118.5	-1.61607	-0.823276	9039.98	0
4107.74	2341.67	1.15514	-2.19667	425.788	0
2141.77	2084.63	0.154775	-1.68529	369.575	0
FastJet_R0.7					
3.20768e+06	2.63136e+06	-0.649477	2.09788	99980.8	31
2.95354e+06	2.57711e+06	0.531247	-1.03211	156348	28
103066	79570.9	-0.705709	-1.66595	24048.3	30
64279.4	23837.4	-1.64215	-0.765054	7134.36	23
4329.48	2519.45	1.12976	-2.16588	452.33	11
2958.39	1146.29	-1.58766	-2.4412	468.107	6
JetClu_R0.7_M0.75					
3.20951e+06	2.63203e+06	-0.649693	2.09814	118132	35
2.95354e+06	2.57711e+06	0.531247	-1.03211	156348	28
96254.3	76356.2	-0.670059	-1.68225	20024.4	23
71091	26872.7	-1.61838	-0.830413	9535.25	30
5768.06	2900.23	1.28918	-2.09085	1095.28	12
2958.39	1146.29	-1.58766	-2.4412	468.107	6
MidPoint_R0.7_M0.75					
3.20757e+06	2.63128e+06	-0.649483	2.09791	99034.9	30
2.9512e+06	2.57508e+06	0.531599	-1.03173	146101	26
95630.3	76008.4	-0.667749	-1.68487	19633.6	21
70143.9	26967.6	-1.59986	-0.835052	9490.66	29
4329.48	2519.45	1.12976	-2.16588	452.33	11
2343.35	2264.18	0.0406512	-1.48787	596.903	2
2958.39	1146.29	-1.58766	-2.4412	468.107	6
SISCone_R0.7_M0.75					
3.20757e+06	2.63128e+06	-0.649483	2.09791	99034.9	30
2.9512e+06	2.57508e+06	0.531599	-1.03173	146101	26
95630.3	76008.4	-0.667749	-1.68487	19633.6	21
70143.9	26967.6	-1.59986	-0.835052	9490.66	29
4329.48	2519.45	1.12976	-2.16588	452.33	11
2343.35	2264.18	0.0406512	-1.48787	596.903	2
2958.39	1146.29	-1.58766	-2.4412	468.107	6

[q]: stop [ENTER]: Continue...

What's Next?

- finish StdHep input classes
- add Y-splitter to break jets into subjets
- more work on stand-alone version
- more documentation?



SpartyJet is in a perpetual state of development; updates and stable versions will be posted on the website.

<http://www.pa.msu.edu/~huston/SpartyJet/SpartyJet.html>