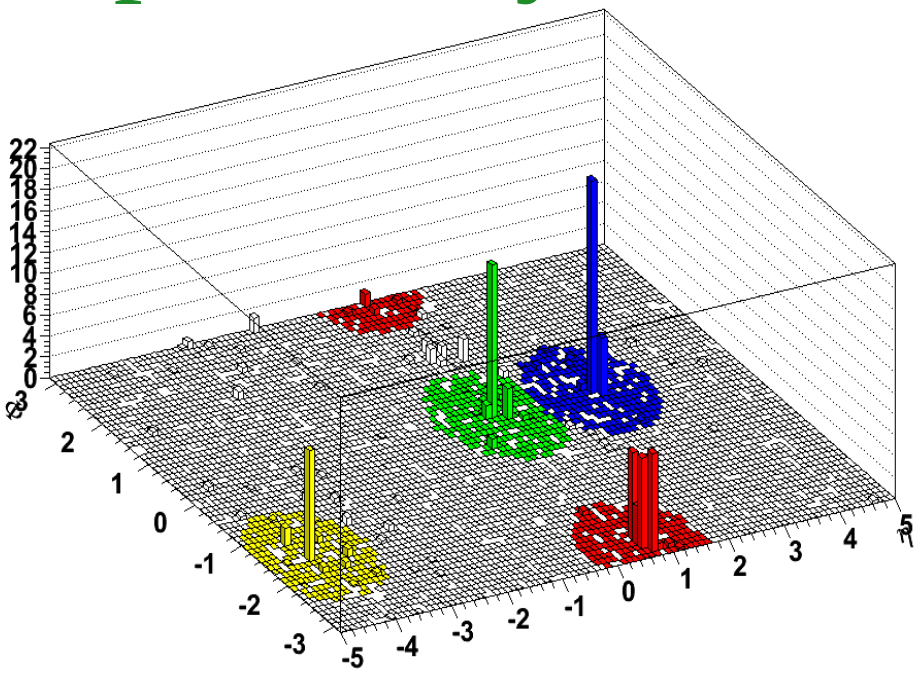
Tier 3 Use Example: D3PD making => JetFinding with SpartyJet=> Ntuple Analysis

Brian Martin

June 8th, 2010

ANL – Tier3 Meeting



Introduction

This is a collection of tasks one might do on a tier 3 for an analysis

- *In fact its pretty much how I use the tier3*

1. Local ATHENA use: Modify/Test EWPA package to produce sample D3PD
2. Pathena use: Submit to GRID using pathena to produce D3PDs for entire dataset
3. DQ2 use: Retrieve output dataset to Tier3.
4. Interactive ROOT: Run ROOT to design/test SpartyJet job.
5. Batch use: Run SpartyJet on entire dataset utilizing bach system (Arcond/Condor)
6. Ntuple use: Final analysis in compiled ROOT code to produce histograms

Exercises much of the Tier3 capabilities

Introduction

Every**W**here**P**hysics**A**nalysis – common,supported DPD making tools, analysis framework

- Very configurable
- Used extensively in ATLAS,

Author: *Massimiliano Bellomo*

Sadly, EWPA has been deprecated in favor of D3PDMaker

SpartyJet – Jet finding interfaces as well as core jet finding.

- Modular Jet Tool design
- Handles multiple types of input
- Native algorithms and built-in FastJet implementation
- Produces convenient ROOT output
- http://projects.hepforge.org/spartyjet/

Authors: *Joey Huston, Pierre-Antoine Delsart*

Kurtis Geerlings, Brian Martin, Chris Vermillion

EWPA Example: D3PD production

Setup Athena:

```
> export ATLAS_LOCAL_ROOT_BASE=/export/share/atlas/ATLASLocalRootBase
> source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh
> export ATLAS_TEST_AREA=/users/brianmartin/testarea_t3g/15.6.7
> localSetupGcc --gccVersion=gcc432_x86_64_slc5
> source /export/home/atlasadmin
        /temp/setupScripts/setupAtlasProduction_15.6.7.sh
```

Enclosed this in a setup script

Get Job options, configure and run:

```
> cp ~brianmartin/public/JamboreeApr2010/makeWjetsD3PD.py .
> vim makeWjetsD3PD.py
> athena makeWjetsD3PD.py
```

SpartyJet Algorithms

- SpartyJet directly implements some algorithms:
  - Atlas Cone&FastKt – CDF MidPoint&JetClu
  - Pythia CellJet – D0 Cone
- SpartyJet links to FastJet to allow use of FastJet algorithms
  - Currently this is setup to allow the native algorithms (kT, anti-kT, and Cambridge-Aachen) plus the SISCone plugin.
  - Many other algorithms can be used via the Plugin interface
  - FastJet Ysplitter interfaced as well
- SpartyJet implements various jet tools as well
  - Jet shapes, filters, Calorimeter Grid, JetArea correction, ...
  - These act on a jet list in succession
    - In SpartyJet a jet algorithm is simply a specific type of jet tool

SpartyJet Architecture

InputMaker

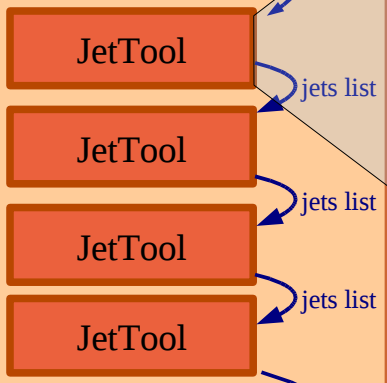
```
fillInput(int eventn,
      Jet::jet_list_t &inputList)
```

Reads an input collection of 4-vectors and converts to an **initial jets list**

JetAlgorithm

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

Feed initial jets list into sequence of JetTools



JetTool modify Jets in sequence

```
execute(JetCollectio
n &inputJets)
```

NtupleMaker

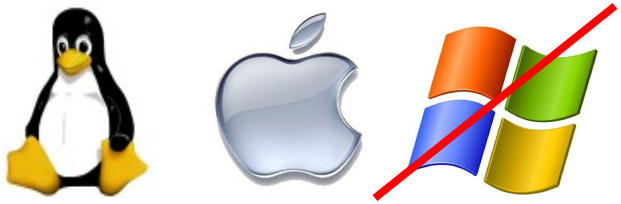
```
addJetVar(std::string jetname);
set_data(std::string jetname,
      JetCollection &theJets);
```

Handles ntuple creation of jet results

MICHIGAN STATE UNIVERSITY

User installation of local software

SpartyJet is now available on HEP Forge



Requirements: linux or mac OSX,

ROOT >=5.18, python >=2.4

- Before compiling on ASC Tier3g:

```
localSetupGcc -gccVersion=gcc432_x86_64_slc5
localSetupPython --pythonVersion=2.5.2
localSetupROOT --rootVersion=5.26.00-slc5-gcc4.3
```

- Get tarball from HEPForge and compile:

```
> wget http://projects.hepforge.org/spartyjet/spartyjet_3.4.1.tar.gz
> tar -xvzf spartyjet_3.4.1.tar.gz
> cd spartyjet_3.4.1
> make
```

Running SpartyJet

SpartyJet ships with a set of examples and input data

- Examples are organized by type:
  - Python scripts: spartyjet/examples_py/ Recommended
  - ROOT scripts: spartyjet/examples_ROOT/
  - Compiled C++ programs: spartyjet/examples_C
- All examples should be run from their directories

C++

```
cd  examples_C/
./simpleExample.exe
```

Python

```
source setup.sh
cd  examples_py/
./simpleExample.py
```

ROOT

```
cd  examples_ROOT/
root -l simpleExample.C
```

SpartyJet Example: run on EWPA D3PD

From /users/brianmartin/public/JamboreeApr2010/spartyExample.py

```
builder = SJ.JetBuilder(SJ.INFO) # Create the job manager: JetBuilder
# INFO describes the message output level

# Configure input ------------------------------
input = createNtupleInputMaker('EWPA.D3PD.root',inputprefix='TopoCluster')
builder.configure_input(input)

# Configure algorithms -------------------------
anti4 = SJ.fastjet.FastJetFinder('AntiKt4',fj.antikt_algorithm,0.4)
builder.add_default_alg(anti4) # Add to the builder

# Configure output-----------------------------
# Optional text output
#builder.add_text_output("simple.dat")
# Ntuple Output
builder.configure_output("SpartyJet_Tree","EWPA.SJ.root");

# Run SpartyJet
builder.process_events(10)
```

Pileup Study: Overlay minbias events

1. Produce EWPA D3PD with pileup (go it on the GRID)

```
localSetupPandaClient
pathena makeMBD3PD.py --inDS
mc09_7TeV.105001.pythia_minbias.merge.AOD.e517_s745_s746_r1098_r1113/ --outDS
user10.BrianThomasMartin.mc09_7TeV.105001.pythia_minbias.merge.DPD.e517_s745_s7
46_r1098_r1113_TEST --noBuild --extOutFile EWPA.D3PD.root --dbRelease
ddo.000001.Atlas.Ideal.DBRelease.v080602:DBRelease-8.6.2.tar.gz --nFiles 1
```

2. Copy D3PD from GRID (in clean shell)

```
export ATLAS_LOCAL_ROOT_BASE=/export/share/atlas/ATLASLocalRootBase
source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh
localSetupDQ2Client
voms-proxy-init -voms atlas
dq2-get
user10.BrianThomasMartin.mc09_7TeV.105001.pythia_minbias.merge.DPD.e517_s745_s7
46_r1098_r1113_TEST
```

3. Use SpartyJet to overlay minbias event on signal events.

```
# Overlay Minbias events
mbInput =
createNtupleInputMaker('/users/brianmartin/public/JamboreeApr2010/Minbias.D3PD.
root', inputprefix='TopoCluster')
numMinBias = 4
builder.add_minbias_events(numMinBias,mbInput,True)
# Bool is whether to draw no. of mb events from poisson
```

Runs on Tier3g Batch using Condor

1. In a clean shell, setup Tier3g and gcc/python/ROOT

```
export ATLAS_LOCAL_ROOT_BASE=/export/share/atlas/ATLASLocalRootBase
source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh
localSetupGcc -gccVersion=gcc432_x86_64_slc5
localSetupPython --pythonVersion=2.5.2
localSetupROOT --rootVersion=5.26.00-slc5-gcc4.3
```

2. Make simple job config file: spartyOnCondor.py

```
universe        = vanilla
executable      =/users/brianmartin/public/JamboreeApr2010/spartyExample.py
getenv          = True
arguments       = ""
output          = job.local.out
error           = job.local.err
log             = job.local.log
WhenToTransferOutput = ON_EXIT_OR_EVICT
queue 1
```

3. Submit to condor

```
condor_submit spartyOnCondor.sub
```

Runs on Tier3g Batch using Arcond

Even more convenient, I can run an ATHENA job and feed the output directly into SpartyJet.

1. Distribute data on tier3 farms using ArCond tools
2. Automatically run ATHENA => SpartyJet in parallel on all nodes

1. Edit ArCond shell script to execute SpartyJet after ATHENA job finishes

```
>vim  user/ShellScript_BASIC.sh
```

2. Collect results and merge for final ROOT analysis

```
>arc_add
################# ArCond-1.6 ####################
##                  ANL ASC                    ##
################################################
 --- Current directory ----
/users/brianmartin/pcfTest
 -------------------- ----
---> File name from each job = Analysis.root
---> File name containing combined output = Analysis_all.root
```

SpartyJet Example: FastJet plugins

From FJExample.py

```
### JetPruning:
antikt10 = fj.JetDefinition(fj.antikt_algorithm,1.0)
antiktBIG = fj.JetDefinition(fj.antikt_algorithm,3.14*0.5)
prunePlugin = fj.FastPrunePlugin(antikt10,antiktBIG,0.1,0.5)
pruneJetDef = fj.JetDefinition(prunePlugin)
prune = SJ.FastJet.FastJetFinder(pruneJetDef,'Prune',False)
builder.add_default_alg(prune)
```

Many exciting new jet sub-structure tools being developed in the community:

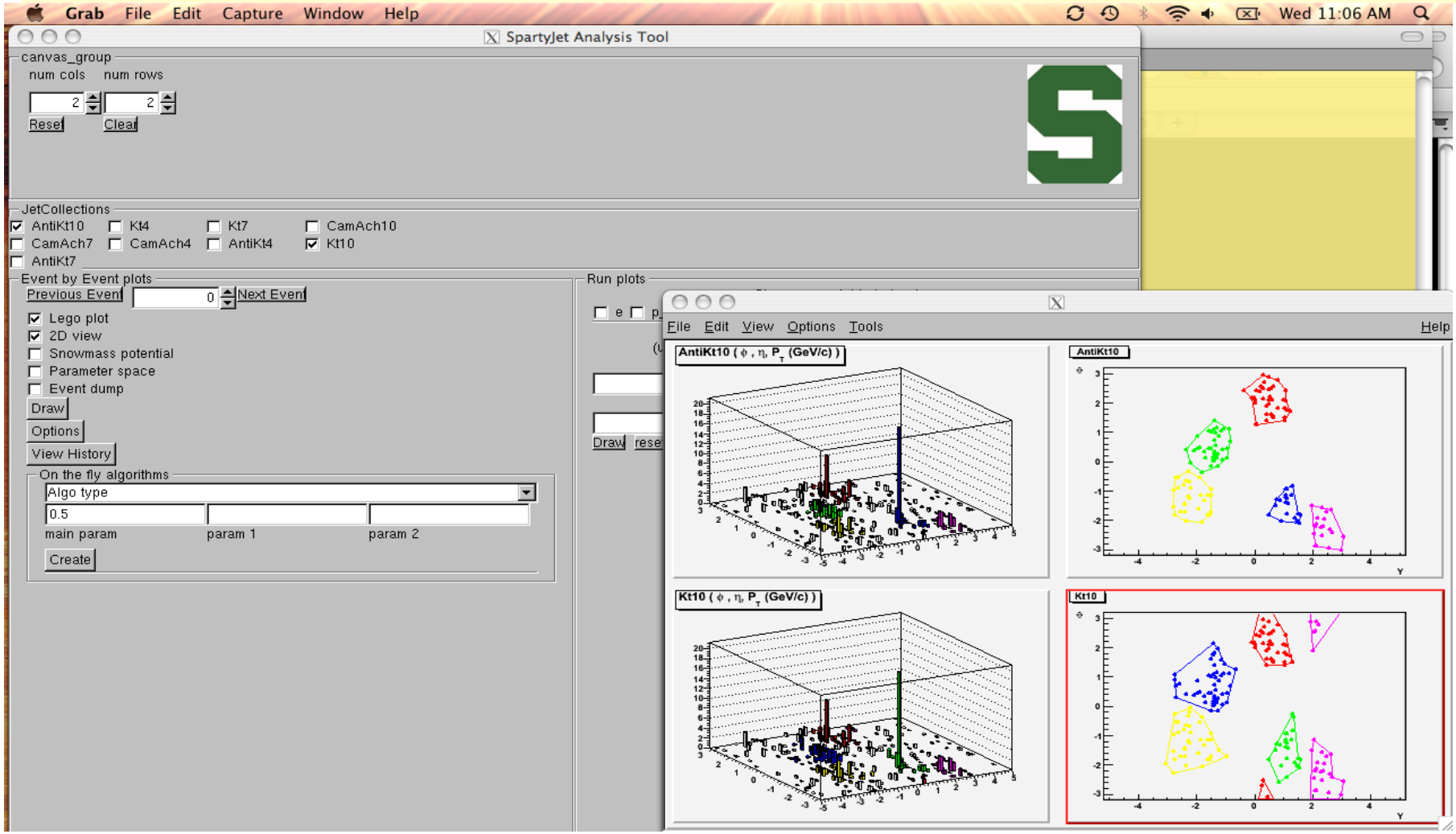
- Jet Pruning:
  http://www.phys.washington.edu/groups/lhcti/pruning/FastJetPlugin/
- Trimming: http://jthaler.net/jets/Jet_Trimming.html
- Variable R jets
- Various taggers

- These plugins can be used directly in SpartyJet

SpartyJet Example: gui (alpha)

guiExample.py (Requires additional X11 lib)

```
./guiExample.py
```



Conclusions

- The Tier3g setup has all the functionality to carry out analysis requiring diverse tools
- It is easy to use and well documented
  - Starting from never having run on this system before, I was able to do all of this in very little time (much less than an hour, including GRID time)
  - It is flexible enough to allow customizing the environment for user-specific applications (SpartyJet, GoodRunsList parsing, etc)
    - On other tier3s this can be quite an issue and require sys admin time
- Most importantly, SpartyJet runs on it.