

# MC validation and tuning tools

## Rivet & Professor

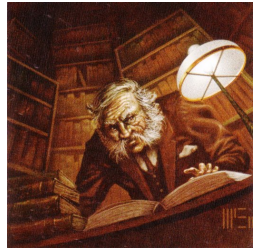
Andy Buckley  
University of Edinburgh

MC4LHC readiness workshop, CERN, 2010-03-31



# Contents

- 1 Rivet
- 2 Professor
- 3 Conclusions



# Rivet and Professor

Rivet and Professor are:

- ▶ Contrived acronyms!
- ▶ Tools for checking and improving generator tunes
- ▶ Tools for improving generator models
- ▶ Available to use *and* contribute towards
- ▶ Used to some extent by all LHC collabs (except ALICE?)

They are not...

- ▶ ...magic!  $\Rightarrow$  Garbage in, garbage out.

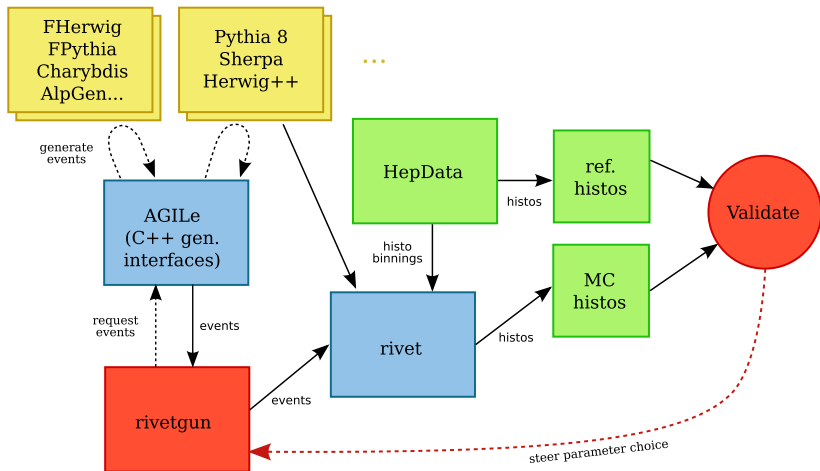
Let's begin.

Rivet

# Rivet at 100 km/h

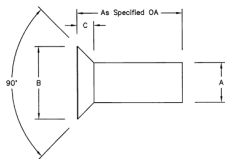
- ▶ Rivet is (to 1st order) “HZTOOL++”
- ▶ Tool for replicating experimental analyses for MC generators
- ▶ With some lessons learnt:
  - big emphasis on **generator independence**
  - $\Rightarrow$  split steering from analysis
- ▶ Tools and key analyses in one system
- ▶ Also: usable as library or executable, dev or user
- ▶ **Current release: 1.2.1, about 1 week old!**

# A complete validation/tuning system



## Some Rivet characteristics

- ▶ Using HepMC pipes for generic processing with `rivet`
- ▶ LWH/AIDA for histogramming: **long-overdue replacement!**
- ▶ All analyses loaded from as runtime “plugins”
- ▶ Reference data bundled... most exported from HepData
- ▶ Code structured with “projections” to cache computations
  - Makes writing analyses very clean and compact
- ▶ “AGILe” gen interfaces for convenience with Fortran gens → HepMC



Designing a Rivet...

# Running Rivet

Use the `rivet` command line tool. Controls which analyses get run, to view analysis metadata, write out histos in several formats, etc.

Metadata:

```
rivet --list-analyses # add "-V" for titles
rivet --show-analysis D0_2008_ # pattern matching
```

Analysing:

```
agile-runmc Pythia6:422 --beams=TVT:1960 -n 200000 \  
  -o - | rivet -a D0_2008_S7662670  
## or use e.g. mkfifo hepmc.fifo...
```

Generator input mostly comes as HepMC ASCII data from a steering program. AGILE interfaces can be used to pass generator params for Fortran gens. For C++ gens use native executables.



# Rivet metadata

```
rivet --show-analysis D0_2008_S7863608
D0_2008_S7863608
=====
```

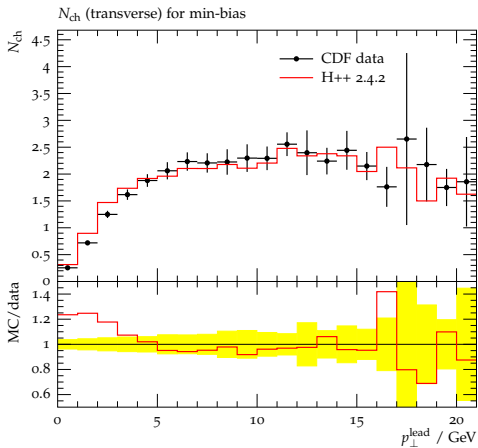
```
Spires ID: 7863608
Spires URL: http://www.slac.stanford.edu/\[...\]=key+7863608
Experiment: D0
Year of publication: 2008
```

```
Description:
  Differential Z/gamma* + jet + X cross sections
```

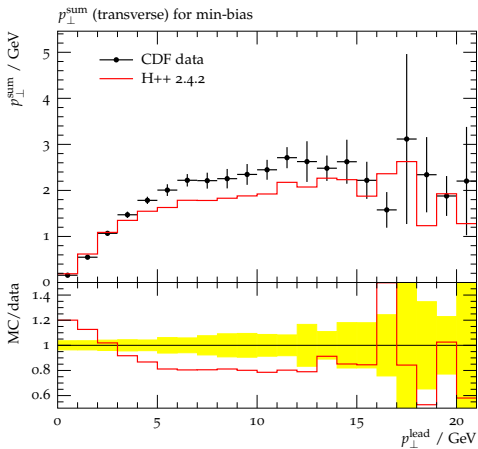
```
References:
  hep-ex/08081296
```

Used to automatically build sections of the manual:  
[arXiv:1003.0694](https://arxiv.org/abs/1003.0694)

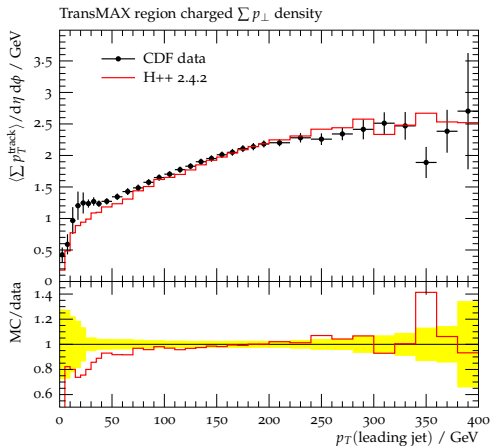
# Some example output



# Some example output



# Some example output



# Getting Rivet

Easiest, if not already installed (e.g. ATLAS, soon CERN TH):  
use [bootstrap script](#)

```
wget http://svn.hepforge.org/rivet/bootstrap/rivet-bootstrap
chmod +x rivet-bootstrap
./rivet-bootstrap
```

Should work on all Linux and Mac platforms: build tested automatically when code changes in repository for many platforms, 64/32 bit, etc..

Bootstrap uses LCG Genser packages when possible (unless forced otherwise).

## Writing an analysis

Easiest way to start: `rivet-mkanalysis` makes template analysis code and metadata files. Might even screen-scrape SPIRES...

Projections registered with a name in the analysis constructor:

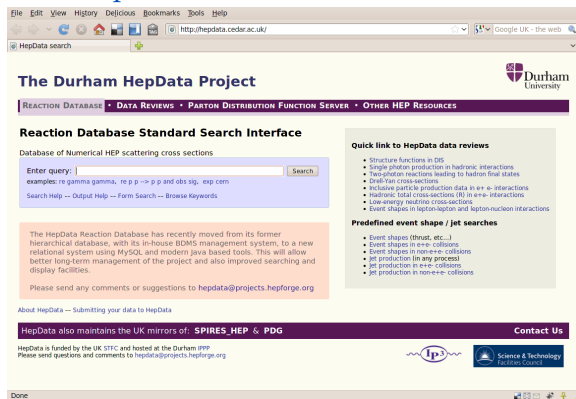
```
class MyAnalysis : public Analysis {
  MyAnalysis() {
    setBeams(PROTON, PROTON);
    ChargedFinalState cfs;
    addProjection(cfs, "CFS");
    ...
  }
};
```

Apply them in the `analyze` method via that name:

```
void MyAnalysis::analyze(const Event& evt) {
  ...
  const FinalState& fs =
    applyProjection<FinalState>(evt, "CFS");
  const ParticleVector ps = fs.particles();
  foreach (const Particle& p, ps) {
    ...
  }
}
```

# Reference data

Bundle reference data for std analyses – mostly obtained **direct** from HepData.



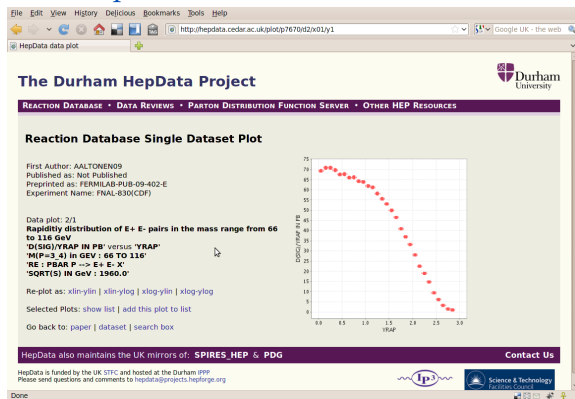
The screenshot shows a web browser window displaying the Durham HepData Project website. The browser's address bar shows the URL <http://hepdata.cedar.ac.uk/>. The website header includes the Durham University logo and a navigation menu with links for REACTION DATABASE, DATA REVIEWS, PARTON DISTRIBUTION FUNCTION SERVER, and OTHER HEP RESOURCES. The main content area features a "Reaction Database Standard Search Interface" with a search box and a "Search" button. Below the search box, there is a text box containing information about the database's recent move to a new system. To the right, there are two sections: "Quick link to HepData data reviews" with a bulleted list of search criteria, and "Predefined event shape / jet searches" with another bulleted list. At the bottom, there is a footer with contact information and logos for ICP3 and the Science & Technology Facilities Council.



MC histograms usually use binnings based on the ref data:  
**automatic consistency.**

# Reference data

Bundle reference data for std analyses – mostly obtained **direct** from HepData.

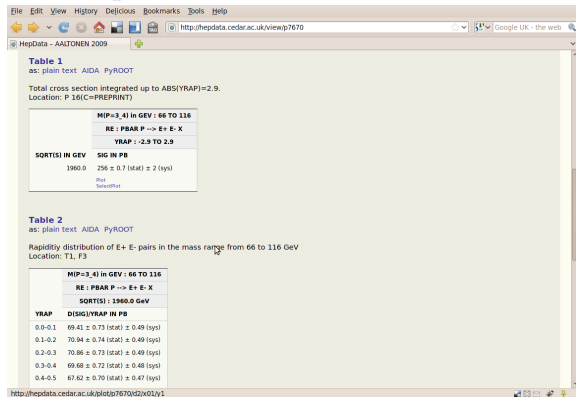


MC histograms usually use binnings based on the ref data:  
**automatic consistency.**



# Reference data

Bundle reference data for std analyses – mostly obtained **direct** from HepData.



MC histograms usually use binnings based on the ref data:  
**automatic consistency.**

# Rivet projections

A quick selection:

- ▶ **Final states:** normal, DIS, “vetoed”, charged, hadronic, unstable (for flavour studies)...
- ▶ **Event shapes:** thrust, sphericity (regularisable), Parisi C & D params, hemispheres...
- ▶ **Jets:**  $k_T$ , CDF “track jet”, DØ ILC, SISCone, CDF RunII Midpoint **all via FastJet**
- ▶ **Misc:** jet shapes, primary vertex position, secondary vertices...

...and lots more. Pretty much everything you need.

## Final Rivet remarks

Rivet is usable for validation, comparison, regression, tuning input... it's quite a generic tool, aided by "small is beautiful"

You can also do evil things with evt record internals if you insist... but not in analyses that we'll take!

Many analyses provided – about 80, including first ATLAS analysis! I believe CMS have a (proposed?) policy that Rivet is used for MC analysis comparisons: awesome! Please supply your experiment's analyses and make your data reproducible forever...

Input your requests – and implementations: BELLE, LEP jet angles, SLD... it would still be nice to have more HERA migration from HZTOOL.

Manpower is an issue: the few core developers can't keep up with the rate of analysis interesting to MC authors. Please help us to keep up, especially if you wrote the data analysis.

Professor

# Parameters

We have lots of parameters:

- ▶ **PS:**  $t_{\min}$ ,  $\alpha_s$  or  $\Lambda_{\text{QCD}}$  (really)
- ▶ **Hadronisation:** depends strongly on model
  - String:** string tension  $\sigma$ , Lund symm FF  $a$  and  $b$  params, baryon suppression, flavour params
  - Cluster:** constituent masses, flavour params
- ▶ **UE:** interaction form factor params (Gaussian width/p(,h)oton radii),  $p_{\perp}^{\min}$ , colour reconnection params
- ▶ **CKKW & friends:** ME/PS matching scale, factorization/renorm. scale

Can sometimes be tuned independently: e.g. kinematics, flavour, UE... depending on analyses

# Tuning methods

Lots of **correlated** parameters,  
**200k–10M events per run** (kin. binning):  
tuning is non-trivial. Too slow for serial  
MCMC sampling approaches to be useful:  
**MC runs are “very expensive functions”.**



**Most tunes:** by eye / by grad student. Painful, uninspiring and sub-optimal. **Hard to repeat!**

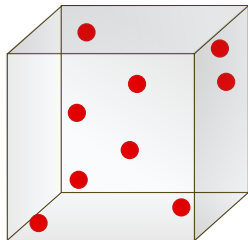
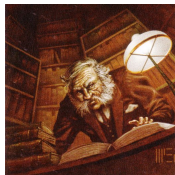
**Herwig++ tunes:** brute-force random on Grid + local param grid scan

**DELPHI:** Hamacher et al (1995) **quadratic interpolation** tune. Scalable, interesting and **it works** ...

**Bear in mind there is some *art* to this...**

# Professor

The **Professor** tuning project (Durham/Edinburgh, Lund/Durham, Berlin) extends the DELPHI approach. Implemented in Python with SciPy (+ **weave**) & PyMinuit.

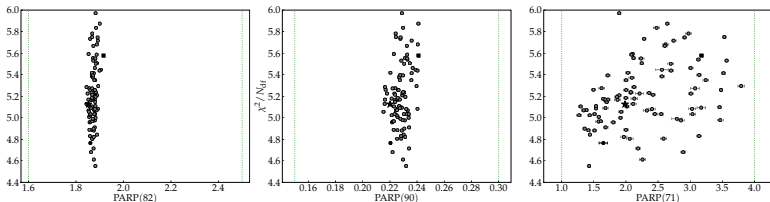


- 1 Sample  $N$  random MC runs from  $n$ -param hypercube using e.g. Rivet
- 2 For each bin  $b$  in each distribution, use the  $N$  points to fit an interpolation function using a singular value decomposition.
- 3 Construct overall  $\chi^2$  function and (numerically) minimise
- 4 Test optimised point by scanning around it in param and lin comb directions

Ask for details... or see the paper:  
arXiv:0907.2973

## Some tune param spreads

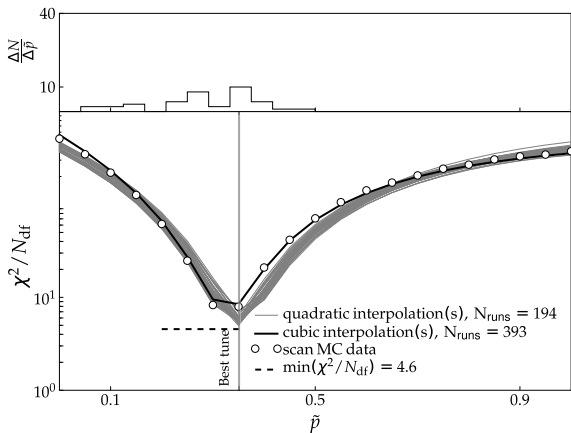
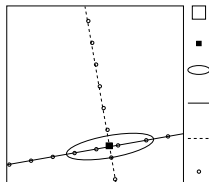
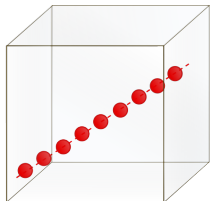
Oversampling required, but if we *really* oversample, then can make many combinations of input MC runs:



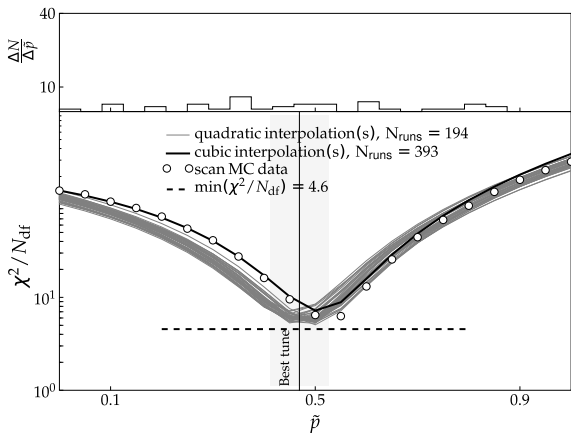
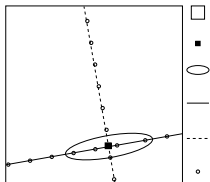
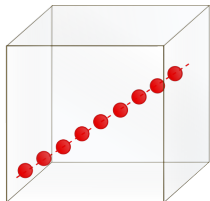
Gives an *informal* picture of how well-constrained (the projection of) a parameter is. Return to this later.



# Checking parameterisation: line-scans



# Checking parameterisation: line-scans



# Tunes

First tunes were “Prof” tunes of PYTHIA 6: much effort in determining parameter factorisations ( $p \lesssim \mathcal{O}(10)$ ) and observable weightings: *the art bit*

Once this was done, easy to make new tunes to other PDFs, etc. with rapid turnaround: length of one high-stats MC iteration:  $\sim 1-3$  days). Extra Prof tunes to CTEQ6L1, LO\*, LO\*\*. Prof tune to LEP fragmentation used in Perugia and other std tunes.

More tunes: Pythia 8 LEP frag tune (model blocks MPI tune so far – see Torbjörn’s talk on Monday), SHERPA new hadronisation tune. SHERPA hadronisation and soft physics development iterated/debugged in close collaboration with Prof tuning: Eike von Seggern and Hendrik Hoeth.

Herwig++ shower proving harder to parameterise: discontinuous parameter behaviours. Will revisit after Professor “hacking week” in late April.

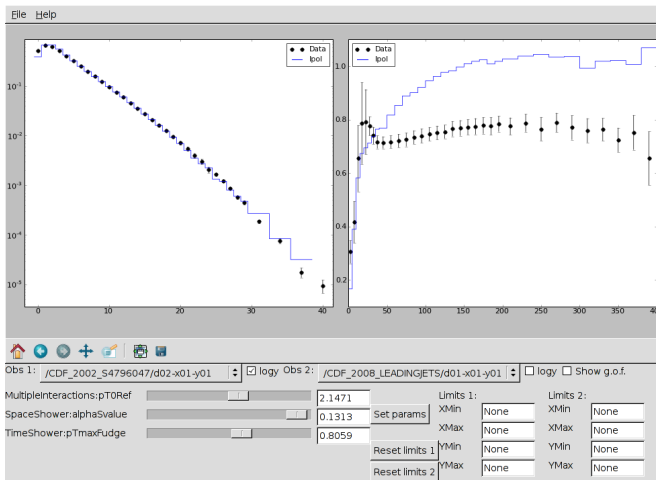
# Interactivity

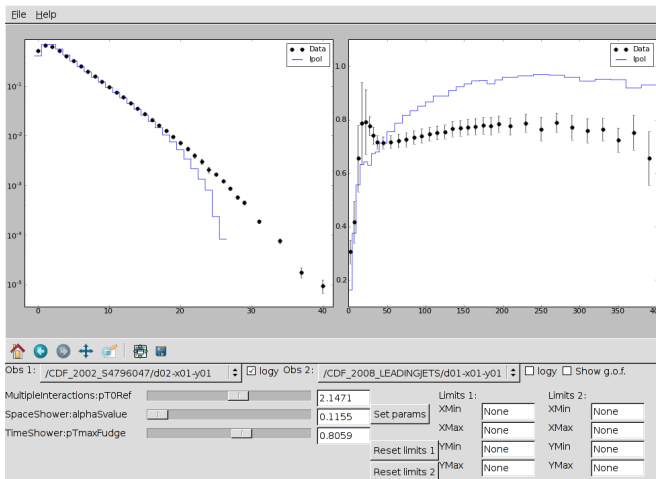
Key feature of Professor (takes time to realise) is that **a) we are parameterising a very expensive function**, and **b) the input to that parameterisation can be trivially parallelised**.

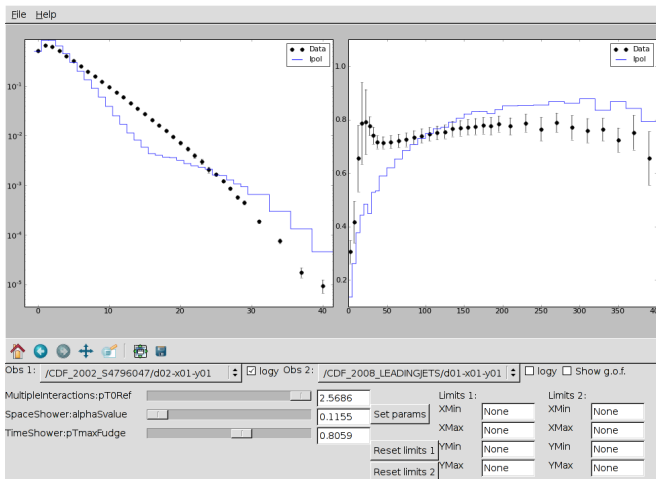
Prof parameterisation (for many, many run combinations) can also be parallelised, as can optimisation.

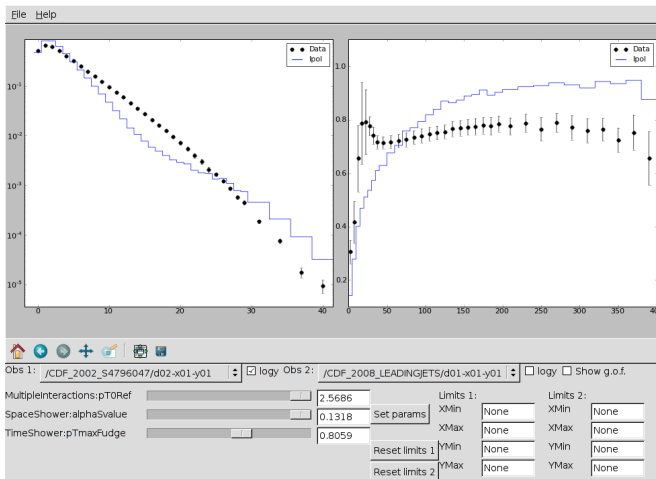
So single-run MC produces a **fast, analytic “pseudo-generator”**. Can get a good approximation of what a generator will do when run for many hours/days with particular params, in  $< 1$  second!

But these things are more general than optimising a tune: why not make an **interactive MC simulator**?

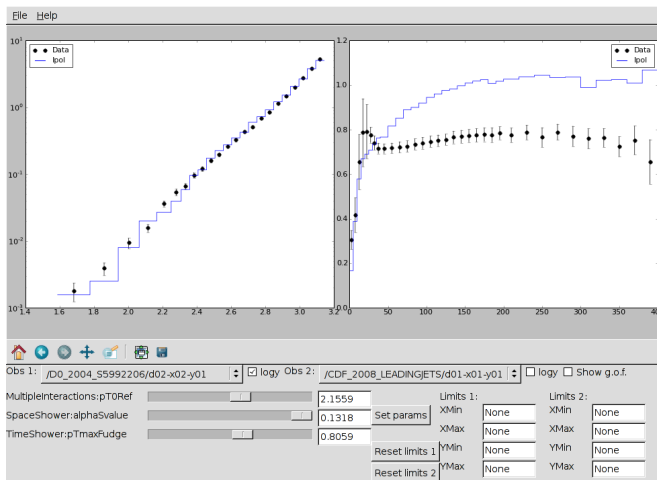


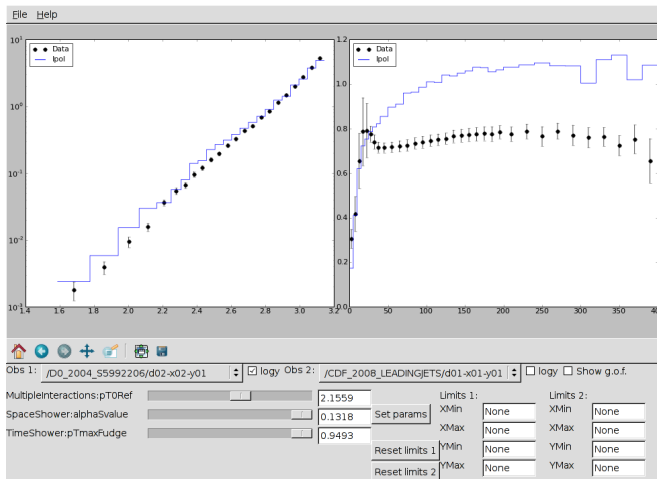






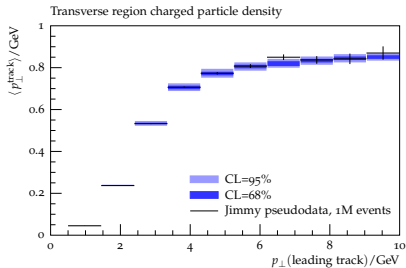
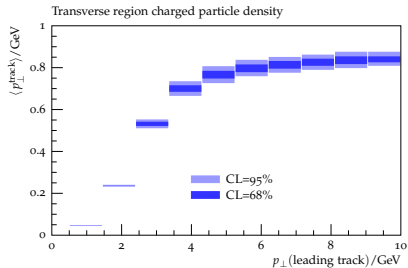






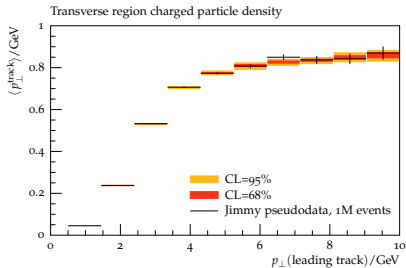
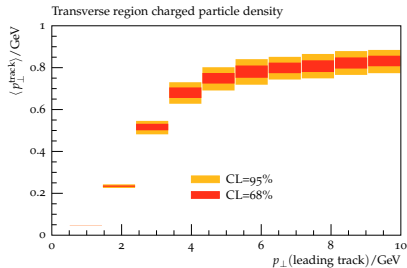
# Statistically-driven tune error bands

Errors from tune ellipsoid sampling



# Statistically-driven tune error bands

Errors from run-combination sampling



# Statistically-driven tune error bands

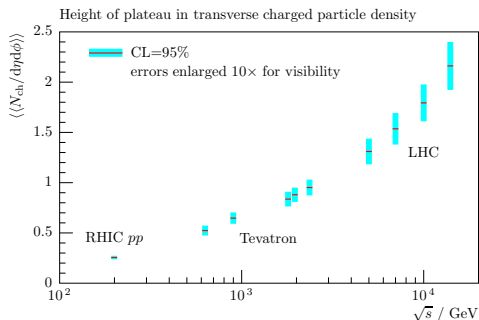
Full errors from combined spread/runcomb sampling



Hopefully coming soon. See the Les Houches 2009/10 tools proceedings: [arXiv:1003.1643](https://arxiv.org/abs/1003.1643)

# Evolution of (partial) tune error beyond measured $\sqrt{s}$

E.g. height of underlying event  $N_{\text{chg}}$  “plateau” vs.  $\sqrt{s}$ :



For PYTHIA in a two-param tune... but not  $p_{\perp}^0$  evolution exponent [PARP(90)], so don't take numbers seriously: it's the principle.

## Final Professor remarks

**Professor works, is public, is increasingly in demand.** ATLAS certainly using it internally / CMS evaluating (?) / some (private?) LHCb use

Quite a few tunes being worked on. **We are now encouraging users to do tunes themselves: we will provide *lots* of help!**

**Tunes are not suddenly a push-button experience: you need to know what the MC can reasonably describe.** Weights not just for whole observables, but even for regions of plots. May need to introduce “meta-parameters” on top of the MC, e.g. relative normalisations or coupled params.

Need to be careful not to “tune away physics”: my feeling is that this has not yet been an issue. **Understanding the MC and the data is key to not falling into the “garbage out” trap!**

# Conclusions



## Summary

Rivet and Professor working well, increasing interest  $\Rightarrow$  lots of demand for us to implement analyses and do tunes... **as for generators, users need to use the tools themselves.** cf. ATLAS

To experimentalists: the point of publication is to produce measurements which stand the test of time and can be used to test new theory/MC models far in the future. **Please write a Rivet routine**, and please, *please* provide data which has been corrected **only for the detector effects**, as well as the model-dependent interpretation, extrapolation etc.. **Not reproducible  $\approx$  useless!**

See Les Houches tools contribution on the issue of  $Z p \perp$  QED and phase-space extrapolation.

We're making an effort to provide latest Rivet and significant Professor snapshots as standard LCG packages: makes it much easier for "average LHC physicist" to use tools.

**In good shape, I think, but there is always room for improvement!**

# Thank you!

A huge thank you to all the CEDAR, MCnet and otherwise-associated people who've contributed to Rivet and Professor over the last couple of years:

Hendrik Hoeth, Frank Siegert, Holger Schulz, James Monk, Gavin Hesketh, Frank Krauss, Eike von Seggern, Emily Nurse, Leif Lönnblad, Lars Sonnenschein, Peter Richardson, Christophe Vaillant, Mike Whalley, Heiko Lacker and more.

Please keep contributing!

