MC generator tuning
Tools and strategy

Andy Buckley
University of Edinburgh

LHCb MC / tuning workshop, Bucharest, 2012-11-23

Introduction

A little (recent) tuning history:

- 2005: A …DW CDF underlying event (+Zp_T, dijet $\Delta\phi$ tunes of PYTHIA6 by Rick Field
- 2008: First "Professor" (global) and "Perugia" (MB) tunes of PY6
- 2009–2011:
  - ATLAS AMBT1–AUET2 tunes of PY6 and HERWIG+JIMMY.
  - Author tunes of Herwig++, Pythia8, and Sherpa
- 2011: ATLAS PY6 AUET2B and Py8 A/AU2 tunes 2012: CMS PY6 $Z2*$, more Sherpa, etc.

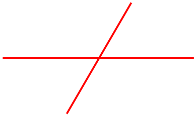
Since 2008 most tuning has made use of Rivet, and often Professor.

This talk: a short intro to these tools, + some opinion/experience on how to approach tuning.

NB. no huge developments in state of the art in the last year – consolidation

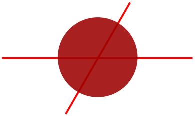
MC generator anatomy (again)

ME (diagrammatic)



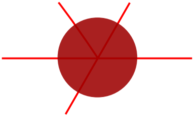
MC generator anatomy (again)

ME (the “right” view)



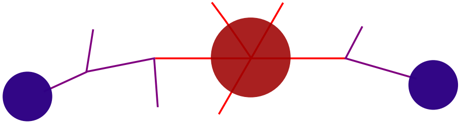
MC generator anatomy (again)

ME (multi-leg)



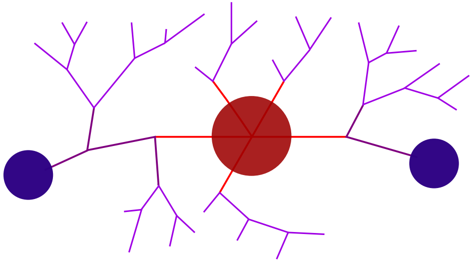
MC generator anatomy (again)

Initial state radiation



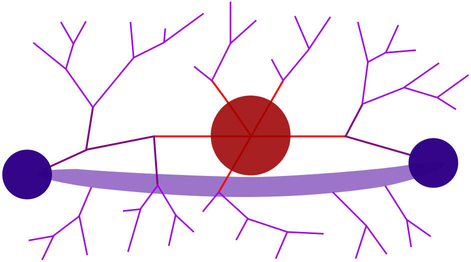
MC generator anatomy (again)

Final state radiation



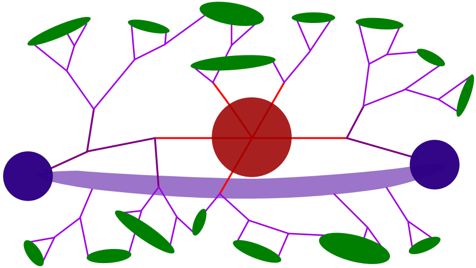
MC generator anatomy (again)

Multiple partonic interactions / UE



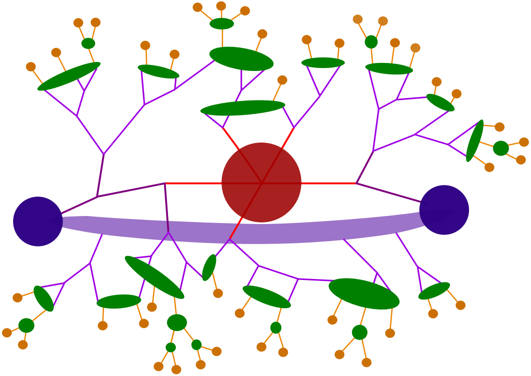
MC generator anatomy (again)

Hadronisation



MC generator anatomy (again)

Hadron decays



What is (allowed for) tuning?

Several approaches to improving generator predictions:

What is (allowed for) tuning?

Several approaches to improving generator predictions:

- put more physics in!
- tuning
- overfitting

What is (allowed for) tuning?

Several approaches to improving generator predictions:

- put more physics in!
- tuning
- overfitting

First two are related: a global tuning that still does badly tells you about model limitations $\Rightarrow$ model improvements.

A substandard tuning of a good model doesn't necessarily tell you anything…

Tuning is necessary for non-perturbative/non-factorisable physics.
Overfitting is bad: it comes when you fine-tune too many knobs and get a good χ^2 but no predictivity.

What is (allowed for) tuning?

Several approaches to improving generator predictions:

- put more physics in!
- tuning
- overfitting

First two are related: a global tuning that still does badly tells you about model limitations ⇒ model improvements.

A substandard tuning of a good model doesn't necessarily tell you anything…

Tuning is necessary for non-perturbative/non-factorisable physics. Overfitting is bad: it comes when you fine-tune too many knobs and get a good χ^2 but no predictivity.

Rule of thumb: avoid tuning perturbative physics as much as possible. Some limited freedom, e.g. μ_R in showers, cf. consistency: most important in merging/matching. *Need systematics coverage.*

Hadronisation and beam remnants are *very* ripe for tuning. Good!

LHCb as a special case

Most MC tuning focused on two types of observable:

- event shapes, jets and identified particle rates and p_Ts at LEP/SLD
- track-based measurements of min bias and underlying event observables at Tevatron and LHC

More recent focus on matched/merged MC. These make tuning harder. IMO shower configs for matching should not be considered as *tuning*.

LHCb phase space and physics focus are unique: great power of EW results for PDFs defn, obvious impact of *b*-physics on general HEP picture.

- multileg and NLO in EW group
- beam remnants and MPI – everywhere, you're a fwd detector!
- fragmentation kinematics and flavour (+ excited states)
- decays – not much tuning room?

Often central tunes do not constrain fwd region much. PDF choice may have a significant impact. Recommend starting from author tunes (NB. use PY6 Perugia2011 for non-default beam remnant treatment)

The tools of the tuning trade: Rivet and Professor

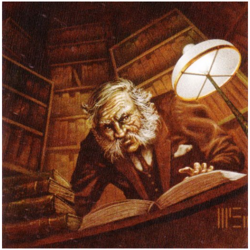
Rivet and Professor are:

- Contrived acronyms. Rivet's is accurate, Professor's misses the mark!
- Tools for checking and improving generator tunes
- Tools for (indirectly) improving generator models
- Available to use *and* contribute towards
- Used to some extent by all LHC collaborations

They are not…

- …magic! ⇒ Garbage in, garbage out.

Let's begin.



Rivet

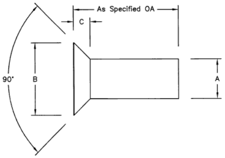
Rivet at 100 km/h

- Rivet is (to 1st order) "HZTOOL++" (for those who remember HZTOOL)
- Tool for replicating experimental analyses for MC generators
- With some lessons learnt:
  - big emphasis on generator-independence. Via HepMC
  - ⇒ split steering from analysis
- Tools and key analyses in one system
- Also: usable as library or executable, dev or user, Python or C++
- **Current release: 1.8.1.** Release 1.8.2 coming *very* soon. 2.0.0 finally near completion: major histogramming overhaul

http://projects.hepforge.org/rivet – includes docs and tutorials.

Some Rivet characteristics

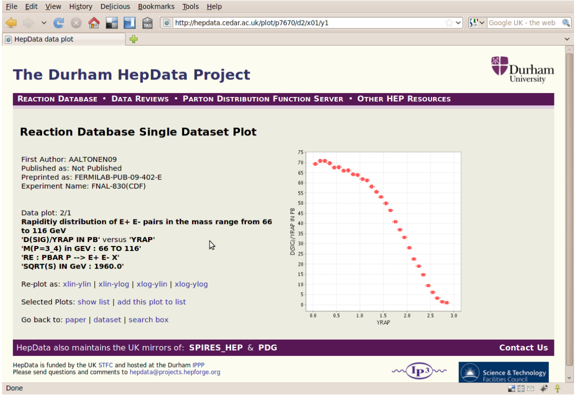
- Recommend HepMC pipes or direct HepMC object passing (faster)
- All analyses loaded from as runtime "plugins"
- Reference data bundled…most exported from HepData
- Code structured with "projections" to cache computations
  - Makes writing analyses very quick, clean, and compact
  - Emphasis on analysis logic over "boilerplate" booking/mangling cruft
  - Try to make default/simplest behaviours the "right" ones, e.g. normalisation & overflows
- "AGILe" gen interfaces for convenience with Fortran gens → HepMC. But we should be mostly using C++ gens these days!



Designing a Rivet…

Reference data

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



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

More ideas like that, please…

Getting Rivet

Easiest is to use the Genser version available on lxplus (same for Professor). Or install your own with the bootstrap script:

```
wget http://rivet.hepforge.org/svn/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).

ATLAS and LHCb (and CMS?) have expt sw interfaces for Rivet.

Rivet 2.0.0 and beyond

- YODA: **http://yoda.hepforge.org** – open SVN trunk to test out / feed back
- YODA features: neat histogram API, bin gaps allowed, fundamental bin width and weight handling, extra weight storage for in-bin distributions and run merging, open to powerful generalisation, C++ and Python interfaces, ...
- Why not use ROOT? *Genuinely* it was going to be too hard to work around all the limitations, and we figured we could do better. Development has been award-deservingly slow, though!
- Next steps: further histogramming extension for parallel weight vector analysis; treatment of correlated NLO counter-events; full run merging machinery; more powerful cuts, decay handling, and jet configuration.
- Aim for main releases up to 2.2.x in 2013.
- Involvement is super-welcome, particularly from LHCb on decay chain analysis etc.

Final Rivet remarks

Rivet has become a de facto standard 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!

However, we're mellowing with age: plan to discuss extension of HepMC status standard and other features/conventions at Les Houches this summer. Some things are not 100% evil and others, like decay chains, deserve better definition.

Many analyses already built in:

```
$ rivet list-analyses | wc l
222
```

!!!

Please supply your experiment's analyses, put the data values in HepData ⇒ key to making your data useful and influential forever…

Professor

Tuning methods

Lots of correlated parameters: shower Λ_{QCD}s and cutoffs; hadronisation string tension, frag functions and flavour tweaks; MPI cutoff, matter overlap, colour reconnection, and energy evolution.

Recommend talking to generator authors about their recommendations. Lots of experience, knowledge, and usually helpful.

200k–10M events per run: tuning is non-trivial.
Too slow for serial MCMC sampling approaches to be useful:
MC runs are "very expensive functions".

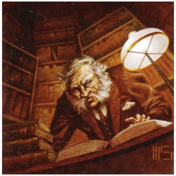
Most tunes: by eye / by grad student. Painful, uninspiring and sub-optimal. Hard to repeat!

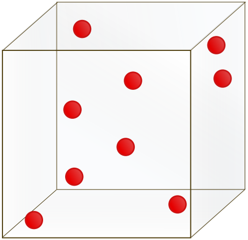
Brute force: hammer the Grid until it hurts.
. . . and the result will still suck.

Parameterise: Hamacher et al (1995) quadratic interpolation tune. Scales beautifully and it works . . . resurrected as Professor

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 χ^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

http://professor.hepforge.org – 1.3.4 bugfix coming soon

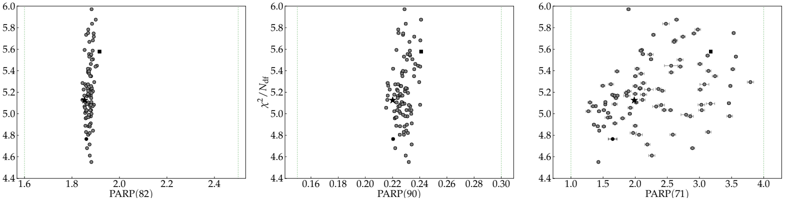
Tuning strategy

Some general idioms to minimise iteration and false steps. These are rules of thumb, not dogma: interesting developments happen when you break the "rules". (But break the right ones!)

- Use as much data as possible – multiple energies, different colliders and detectors – and weight the fits to favour the data you need to describe. Iterate the weights (how?!?)
- Avoid data your model has no chance of describing, e.g. multijets in PYTHIA, diffraction-dominated regimes, etc.
- Factorise the parameters into tuning blocks
- Tune first at e^+e^- to constrain hadr and FSR without initial state hadron complexities: excited states
- Then with a solid base, tune to hadron collider semi-hard QCD. CAREFUL! Shower effects, preferably FSR, e.g. jet shapes.
- Then MPI: models are sufficiently unpredictive that we need to get everything else right first! Super-low p_T needs model++?
  - Diffractive flux model freedoms in Py8? Pile-up or UE tune?
- Matching/merging: need separate matching tunes? Hope not!
- The step yet untaken: return to flavour at hadron colliders.

Some tune param spreads

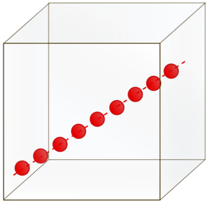
Oversampling required, but if we *really* oversample, then can make many combinations of input MC runs:

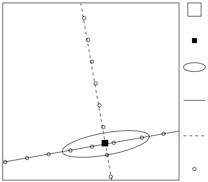


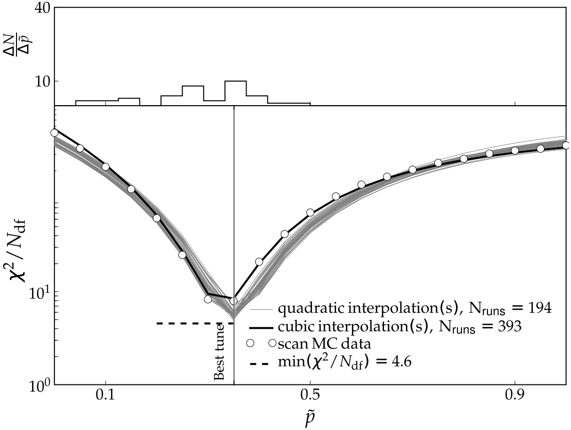
Gives an *informal* picture of how well-constrained (the projection of) a parameter is.

A more correlated but complex version of this, for systematics, can be obtained using the "terror" method and eigentunes.

Checking parameterisation: line-scans







Interactivity

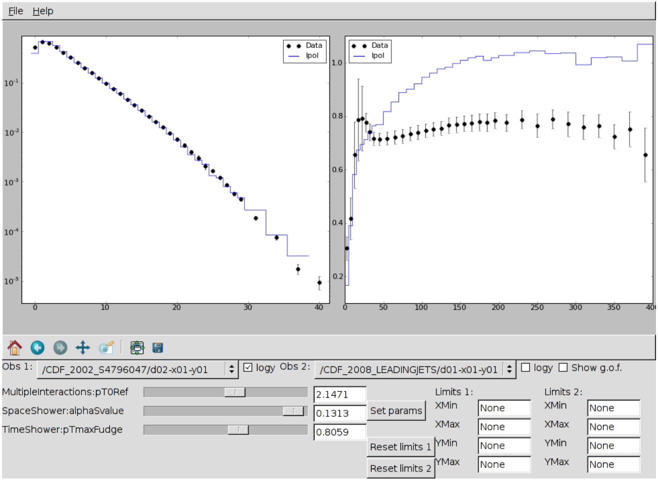
Key feature of Professor (takes time to realise) is that **a)** we are parameterising a very expensive function, and **b)** the input to that parameterisation can be trivially parallelised.

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

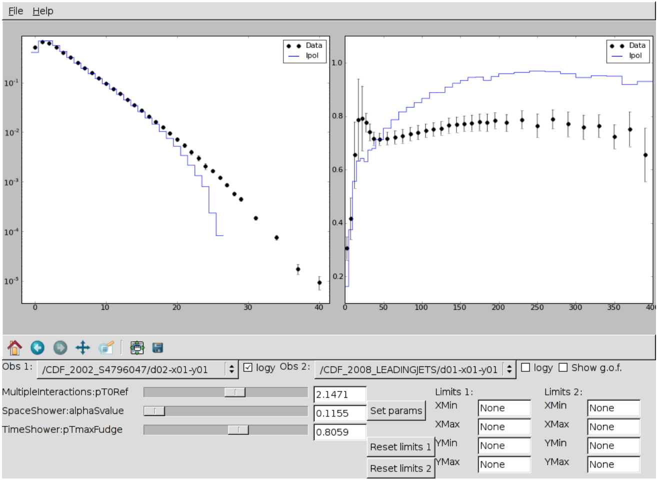
So single-run MC produces a fast, analytic "pseudo-generator". Can get a good approximation of what a generator will do when run for many hours/days with particular params, in < 1 second!

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

prof-I


File
Help
Data
Ipol
Data
Ipol
Obs 1:
/CDF_2002_S4796047/d02-x01-y01
logy
Obs 2:
/CDF_2008_LEADINGJETS/d01-x01-y01
logy
Show g.o.f.
MultipleInteractions:pT0Ref
2.1471
SpaceShower:alphaSvalue
0.1313
TimeShower:pTmaxFudge
0.8059
Set params
Reset limits 1
Reset limits 2
Limits 1:
XMin
None
XMax
None
YMin
None
YMax
None
Limits 2:
XMin
None
XMax
None
YMin
None
YMax
None

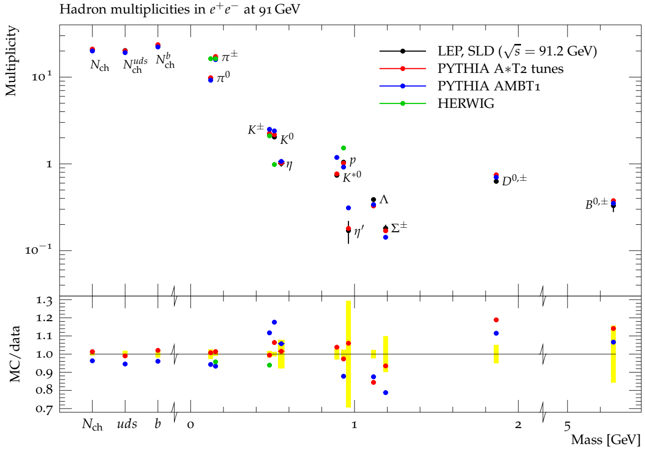
prof-I


File Help
Data
Ipol
Obs 1: /CDF_2002_S4796047/d02-x01-y01
logy
Obs 2: /CDF_2008_LEADINGJETS/d01-x01-y01
logy
Show g.o.f.
MultipleInteractions:pT0Ref
2.1471
SpaceShower:alphaSvalue
0.1155
TimeShower:pTmaxFudge
0.8059
Set params
Reset limits 1
Reset limits 2
Limits 1:
XMin None
XMax None
YMin None
YMax None
Limits 2:
XMin None
XMax None
YMin None
YMax None

Tuning examples

Plot mainly from ATL-PHYS-PUB-2011-008

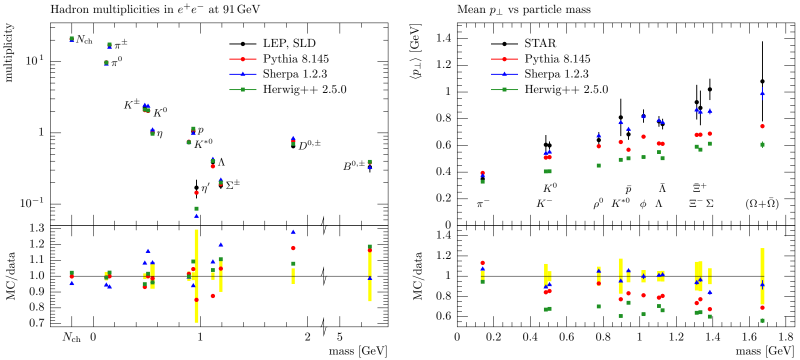
LEP flavour rates and p_T correlations



Tuning examples

Plots from MCnet review (arXiv:1101.2599)

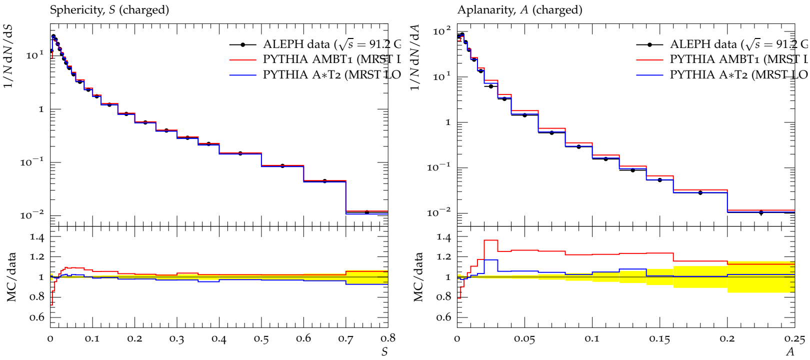
LEP flavour rates and p_T correlations



Tuning examples

Plots from ATL-PHYS-PUB-2011-008

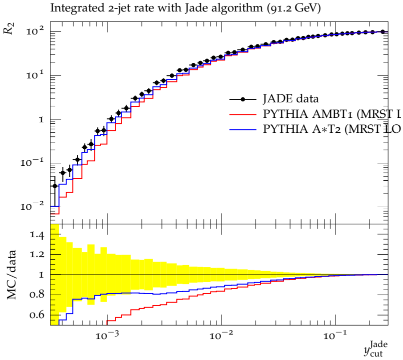
LEP1 event shapes (for PYTHIA p_T-ordered shower – untuned by default. Q^2-ordered is ok.)

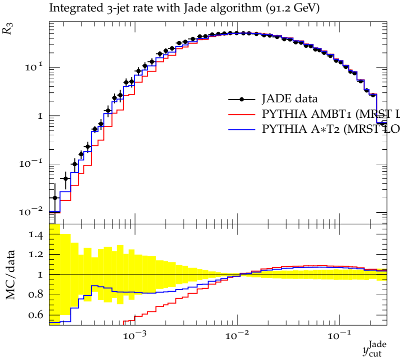


Tuning examples

Plots from ATL-PHYS-PUB-2011-008

LEP/JADE jet structure (for PYTHIA p_T-ordered shower – untuned by default. Q^2-ordered?)

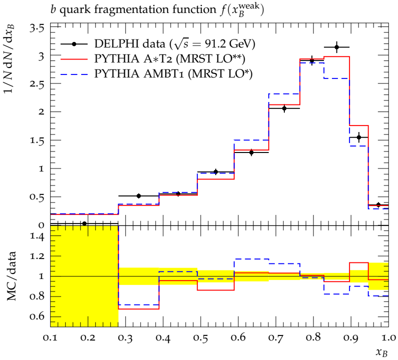


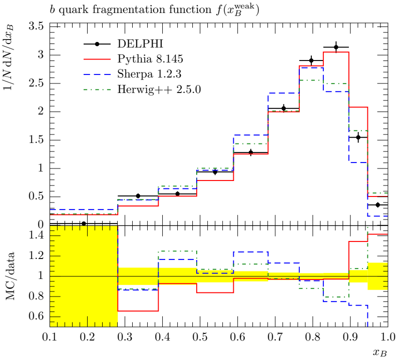


Tuning examples

Plots from ATL-PHYS-PUB-2011-008 and MCnet review (arXiv:1101.2599)

b quark fragmentation function (pre-decay)

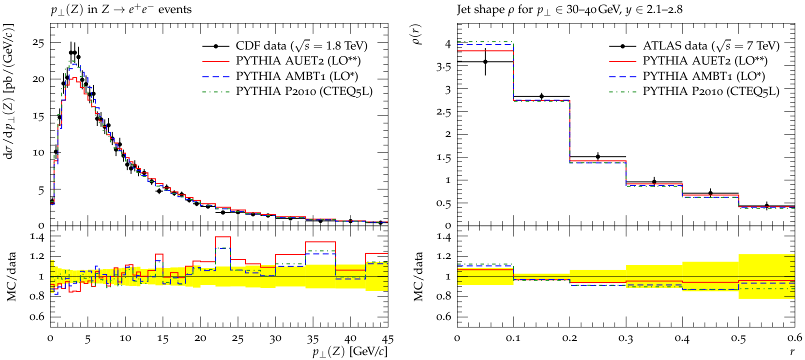




Tuning examples

Plots from ATL-PHYS-PUB-2011-008

Hadron collider ISR: $Z\ p_T$ and intra-jet transverse p_T density ("jet shapes")

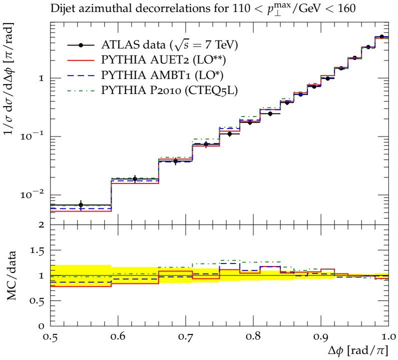


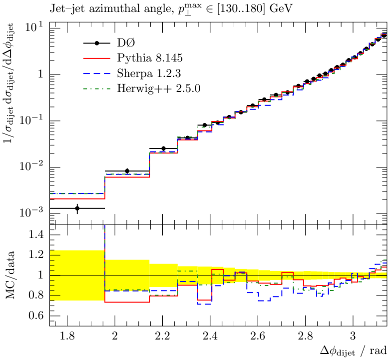
Experience with PYTHIA6 beam remnant – ISR treatment important for universality $\Rightarrow t\bar{t}$

Tuning examples

Plots from ATL-PHYS-PUB-2011-008 and MCnet review (arXiv:1101.2599)

LHC ISR: inter-jet correlations. Careful – much of this should be modelling rather than tuning!

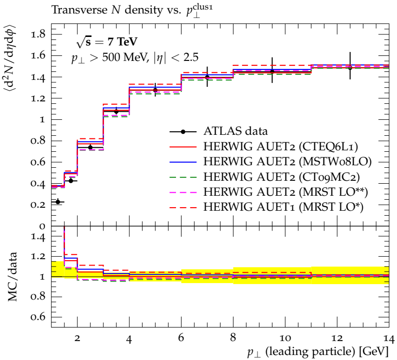


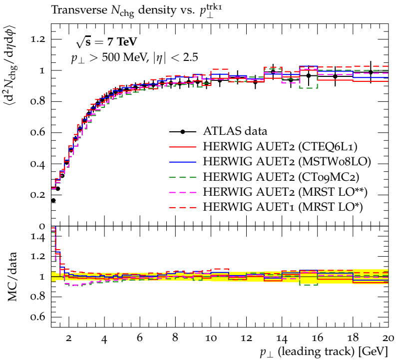


Tuning examples

Plots from ATL-PHYS-PUB-2011-008

Underlying event. JIMMY only – PYTHIA tunes from ATLAS and CMS. Re. C++ gens: Py8 has a flag to damp diffractive cross-sections, H++ gives nice descr.



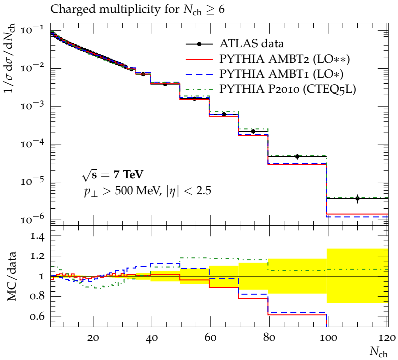


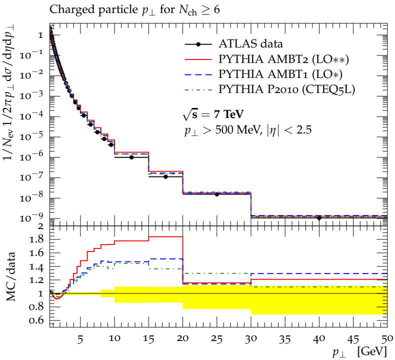
"There is no such thing as the underlying event" – R. Field, MPI@LHC, 2008

Tuning examples

Plots from ATL-PHYS-PUB-2011-008

Min bias. PY6 only. Re. C++ gens: Py8 has a flag to damp diffractive cross-sections, H++ now gives nice description (required CR model)





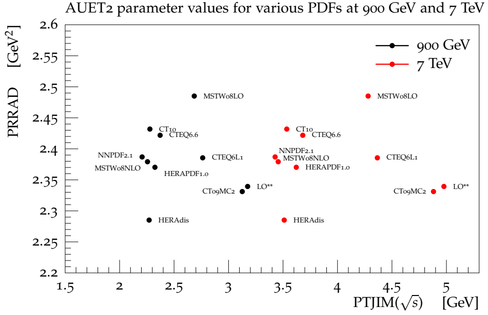
Note that this big bump on the LO** min bias p_T spectrum apparently can't be tuned and depends on the PDF. PDFs also affect UE turn-on shape.

$\Rightarrow$ **Stick with pure LO. What to do with NLO MEs?!?**

Tuning examples

Plot from ATL-PHYS-PUB-2011-008

Automation allows for better systematics via MPI tunes for many PDFs – JIMMY was tuned for 10 PDFs. Note the PDF groupings in p_T^0!



Summary

This is certainly time to switch to the C++ generators, although PYTHIA6 still has a bit of life. Most physics is better there.

Matching generators at NLO, multiple real emissions ("multileg") and *both* is the current trend for high-p_T physics. Other than EW physics, is there a need for NLO/multileg at LHCb? Certainly reduces MC systematics…

Heavy flavour physics in generators still has development room:

- Finite mass treatments hard – but now becoming normal in shower and ME. Also cause trouble in MPI and PDFs
- Improvements in decays, fragmentation, etc.: play a role, please!

Forward η region and LHC identified particles need work: little model constraint, little constraining power from ATLAS & CMS. Effects of PDFs, remnant treatments, etc.: lots of physics

Remember, it's not all tuning: 90% of perturbative MC physics is good model setups. But in the non-perturbative/pheno areas, there is plenty of room for exploration