The Art and Science of Monte Carlo Tuning

Hendrik Hoeth
(Lund University)

Overview

Motivation – "why tuning" and "what's the problem"

Strategy – "how to tune"

Tunings – "plots, plots, plots"

Outlook – "where we are" and "where to go next"

Motivation for Tuning

All generators are based on phenomenological models: string fragmentation, dipole cascade, cluster hadronisation, . . .

The models have free parameters which are a priori unknown and need to be tuned: flavour ratios, q_0^2, fragmentation parameters, intrinsic k_T, . . .

Even parameters like α_s need to be optimised.

We want the MC to have predictive power, so we need to fit as wide a range of available data as possible!

Problems

The model parameters are highly correlated
$\Rightarrow$ can't be tuned one after the other.

Many parameters to be tuned $(\mathcal{O}(10))$.

Tuning all parameters at the same time puts us into a high dimensional parameter space.

Brute force approaches don't work: Running the MC generator takes too long for every point in the parameter space
($=$ setting of parameters).

We haven't the money, so we've got to think.

– Lord Rutherford

Divide and conquer:

Split the task into parts (parton shower, hadronisation, UE)
$\Rightarrow$ cut down the number of parameters.

Be lazy:

Predict the MC output for any parameter set.

Outlining the strategy

1. Choose a tuning interval for the parameters, then pick random points in parameter space and run the generator with these settings.

2. Interpolate between points $\Rightarrow$ prediction of the MC output at any specific parameter setting.

3. Fit this prediction to data (minimal χ^2).

4. Repeat the fit for different combinations of observables.

5. Choose the nicest set of parameters.

(already described and used in Z. Phys., C 73 (1996) 11–59)

1. Choosing parameters

Pick the parameters you want to tune:

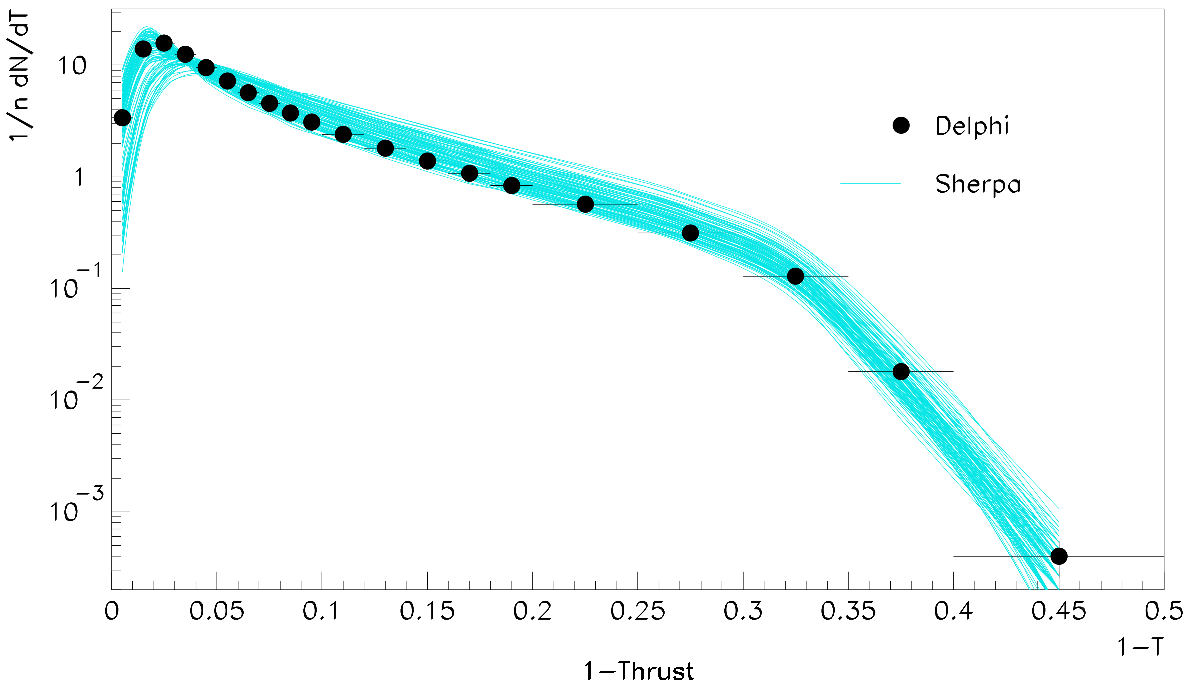
- Tune everything that is important.
- But remember: Each additional parameter adds one dimension to the parameter space.

Define parameter intervals:

- Make the interval large enough so that the result will not be outside.
- But remember: Cutting down 10 intervals by 10 % shrinks the volume of the parameter space by 2/3.

Now pick random points in parameter space and run the generator for each setting.

Calculating observables yields plots like this:



Every line corresponds to a certain parameter setting.

2. Predict the Monte Carlo

Get a bin by bin prediction for the MC response as function of the parameter set $\vec{p} = (p_1, p_2, \ldots, p_n)$.

Interpolate between the parameter points using a order polynomial:

$$MC^{(b)}(\vec{p}) \approx f^{(b)}(\vec{p}) = \alpha_0^{(b)} + \sum_i \beta_i^{(b)} p_i + \sum_{i \leq j} \gamma_{ij}^{(b)} p_i p_j$$

This takes the correlations between the parameters into account.

3. Fit the prediction to data

Using the interpolation we can predict the MC output for any set of parameters very fast. This prediction can be fitted to data, minimising the χ^2:

$$\chi^2(\vec{p}) = \sum_{\text{observables}} \sum_{\text{bins}} \frac{(X_{\text{data}} - X_{\text{MC}}(\vec{p}))^2}{\sigma^2_{\text{data}} + \sigma^2_{\text{MC}}}$$

Include all the relevant data distributions in the fit!

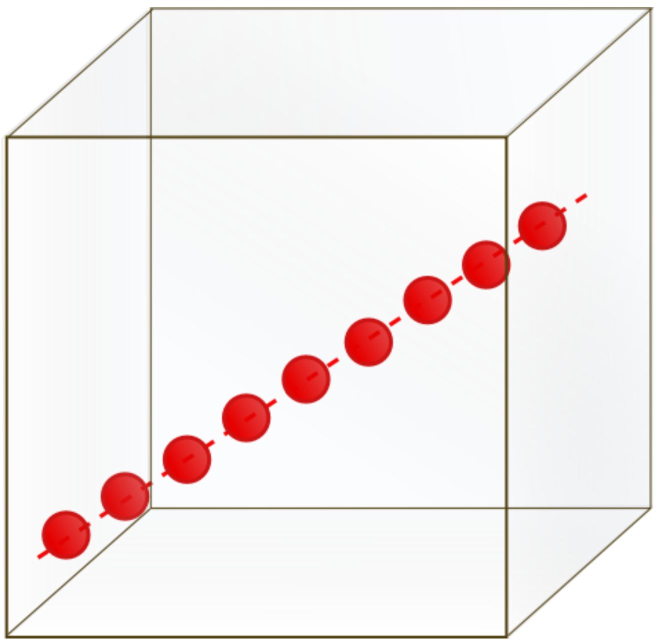
This fit only takes seconds or minutes (as compared to weeks or months for a brute force approach).

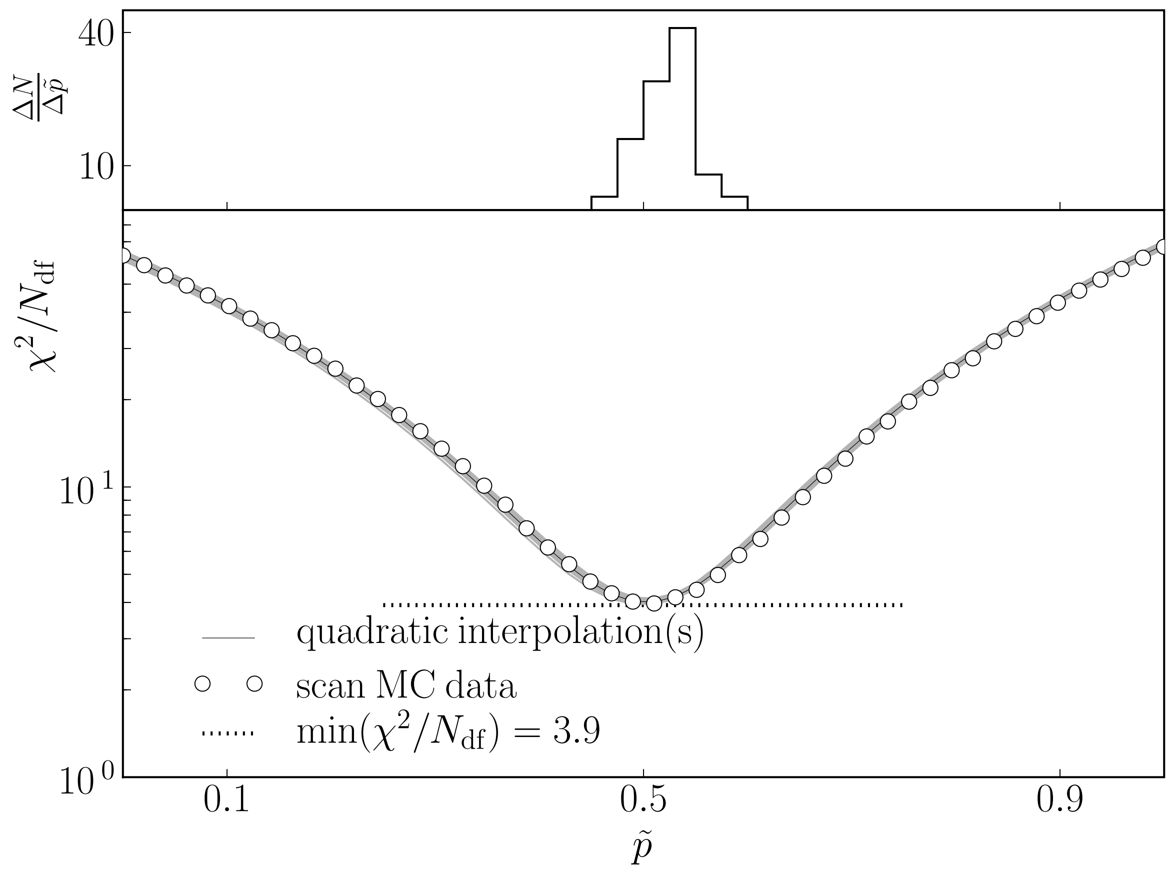
4. + 5. Use different data sets, pick nicest tune

Now we approach the artistic part:

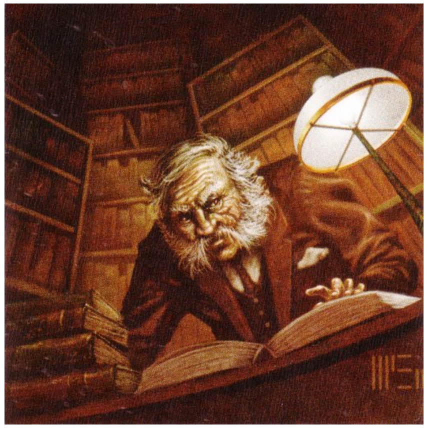
- Use different combinations of observables.
- Put different weights on the observables.
- Learn something about correlations and stability of the tuning.
- Interpret the results in the model's context.
- Maybe adjust/fix parameters by hand.
- Pick the nicest result.

Verifying the Interpolation





Professor and Rivet



Tools for MC tuning have been developed and tested as part of the MCnet programme.

Rivet/(Rivetgun): A general tool to steer different MC generators in a common way and to run analyses on generator level. Lots of published analyses are implemented, direct data/MC comparison is very easy. (http://projects.hepforge.org/rivet/)

Professor: Implementation of the tuning procedure. Uses Rivet to fill histograms. (http://projects.hepforge.org/professor/)

Pythia 6 – New Tunings: LEP

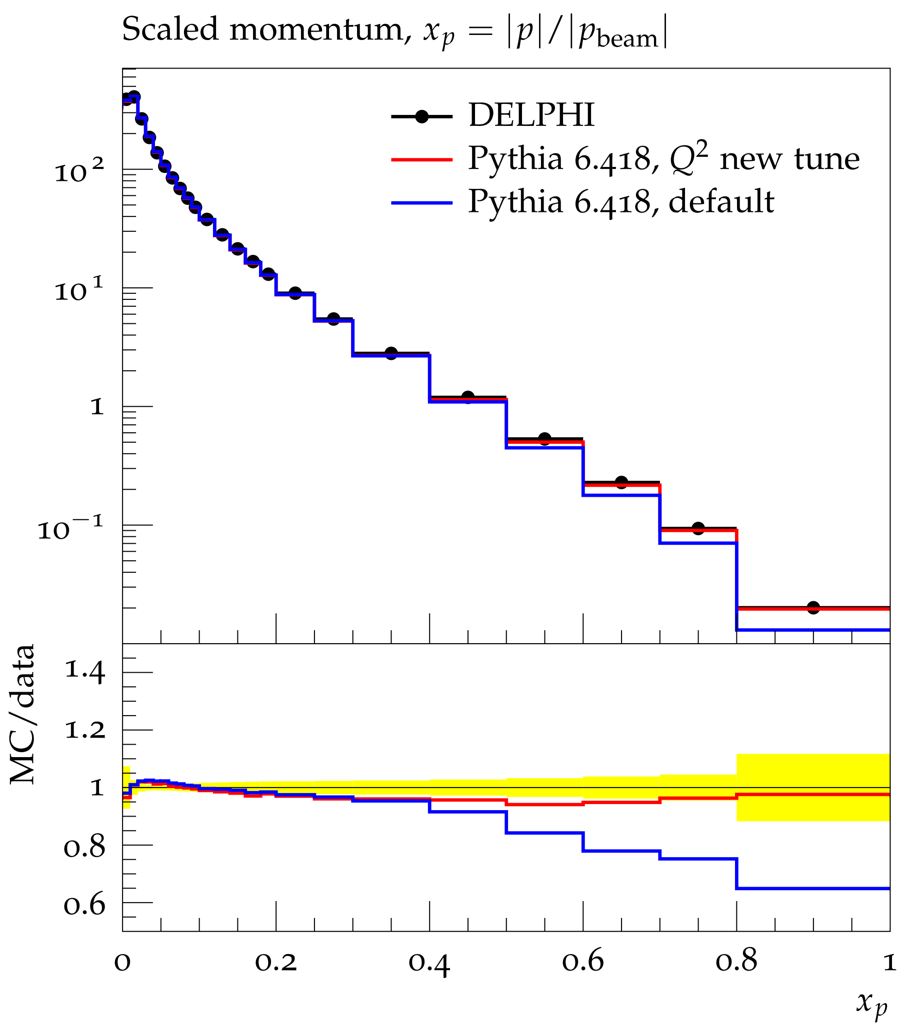
Two-stage tune of Pythia 6 to LEP/SLD data, done for both showers:

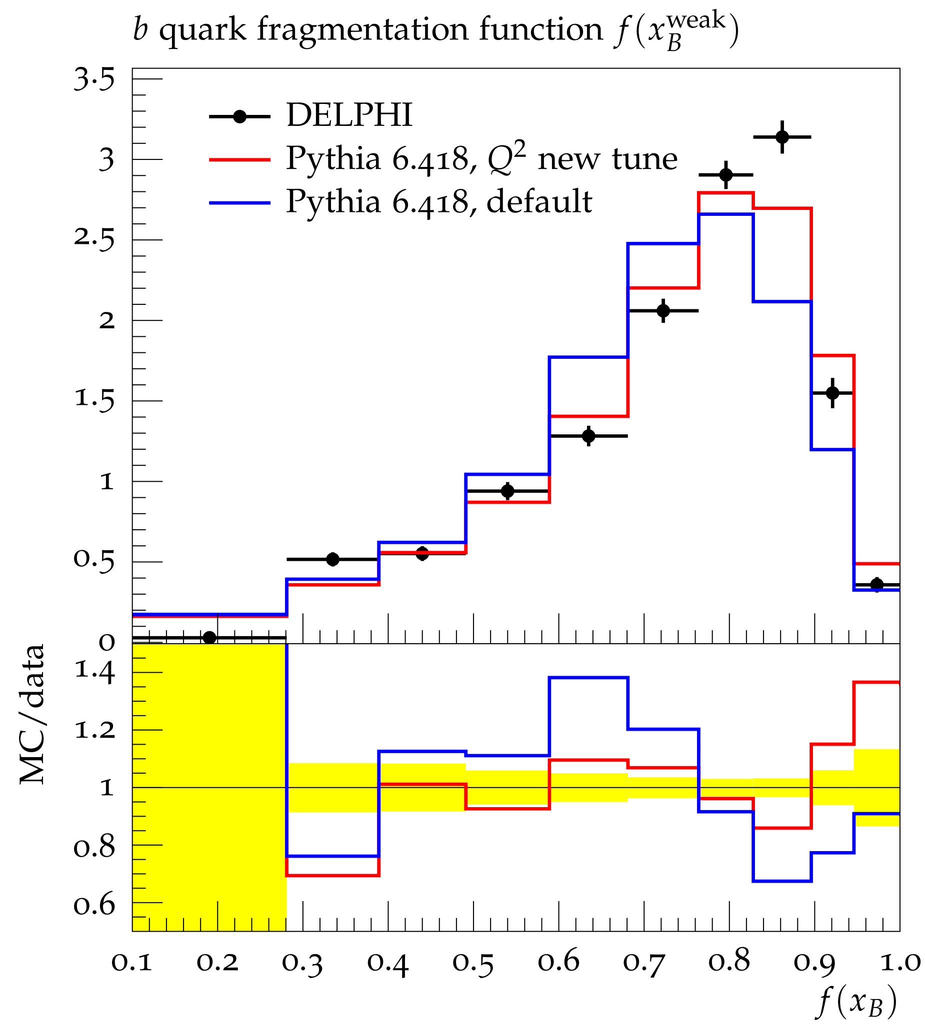
- Flavour parameters. Tuned to identified particle multiplicities, normalized to pions.
- Fragmentation, hadronization. Tuned to event shapes, b-fragmentation measurement, multiplicities, momentum spectra.

Improvement of many identified particle multiplicities and event shapes.

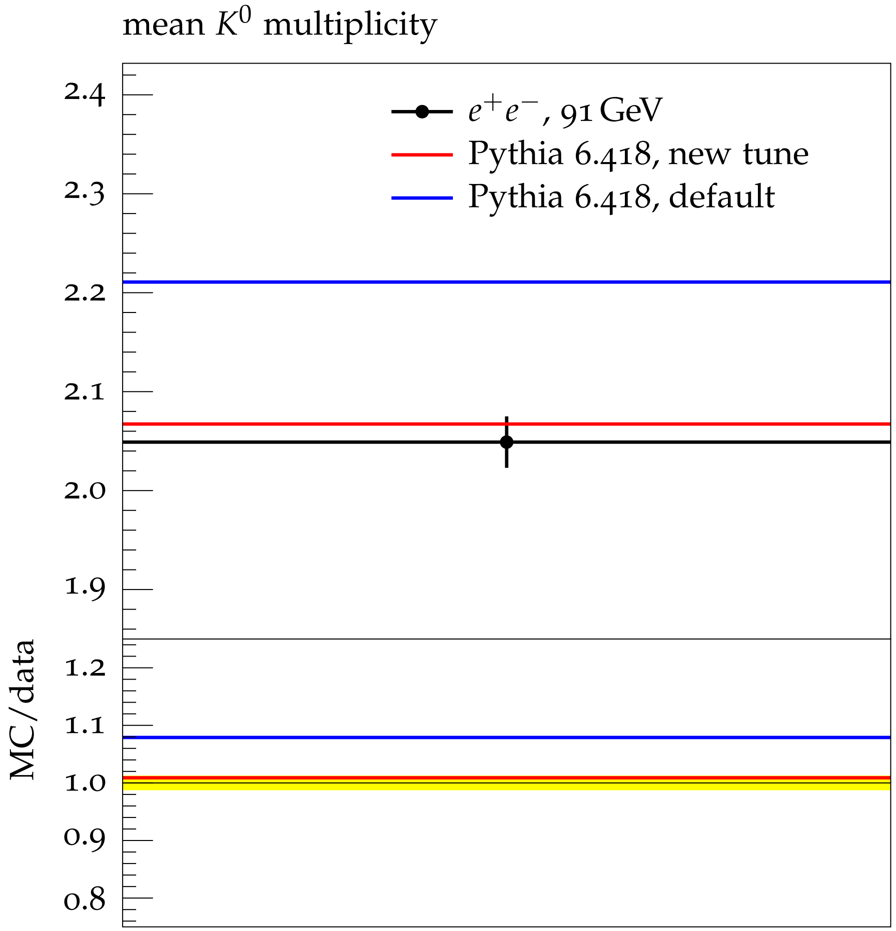
NB: After tuning to LEP data even the agreement with Tevatron data has improved!

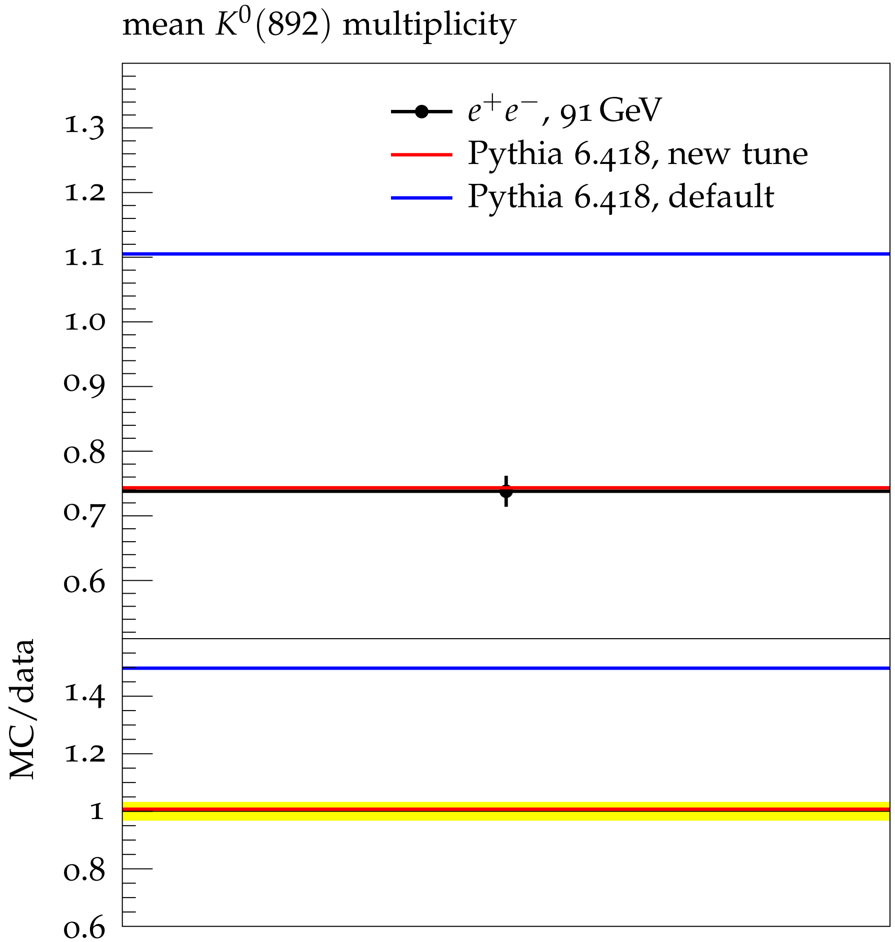
Pythia 6 – New Tunings: LEP



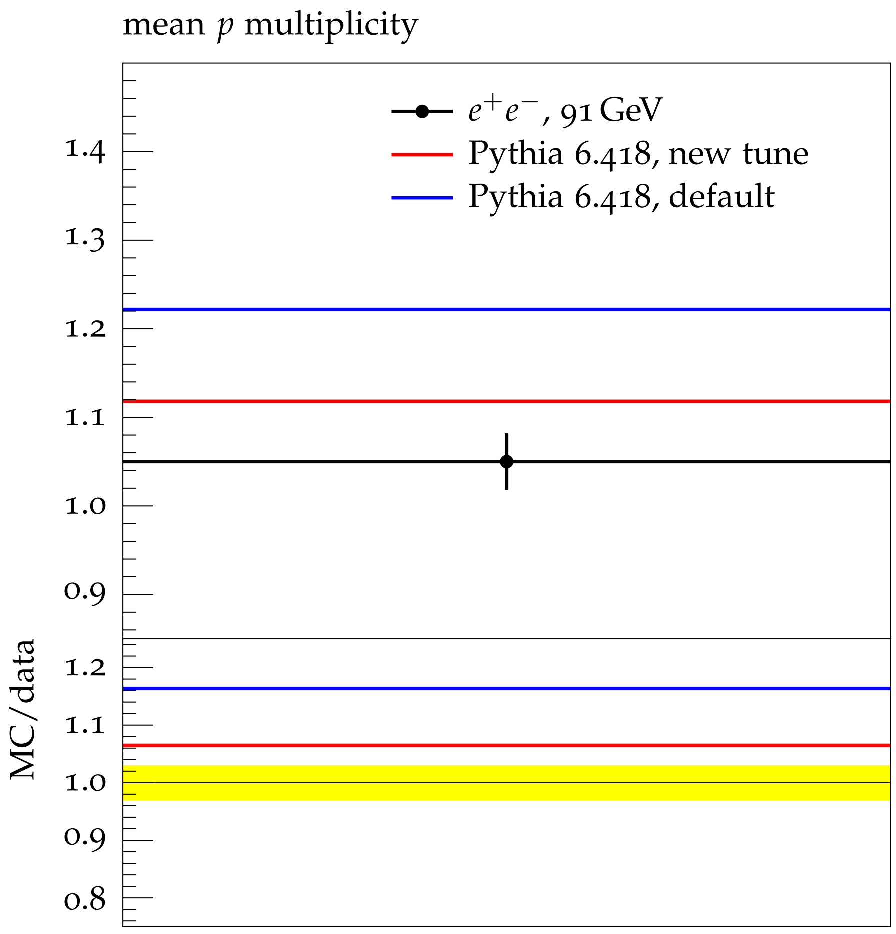


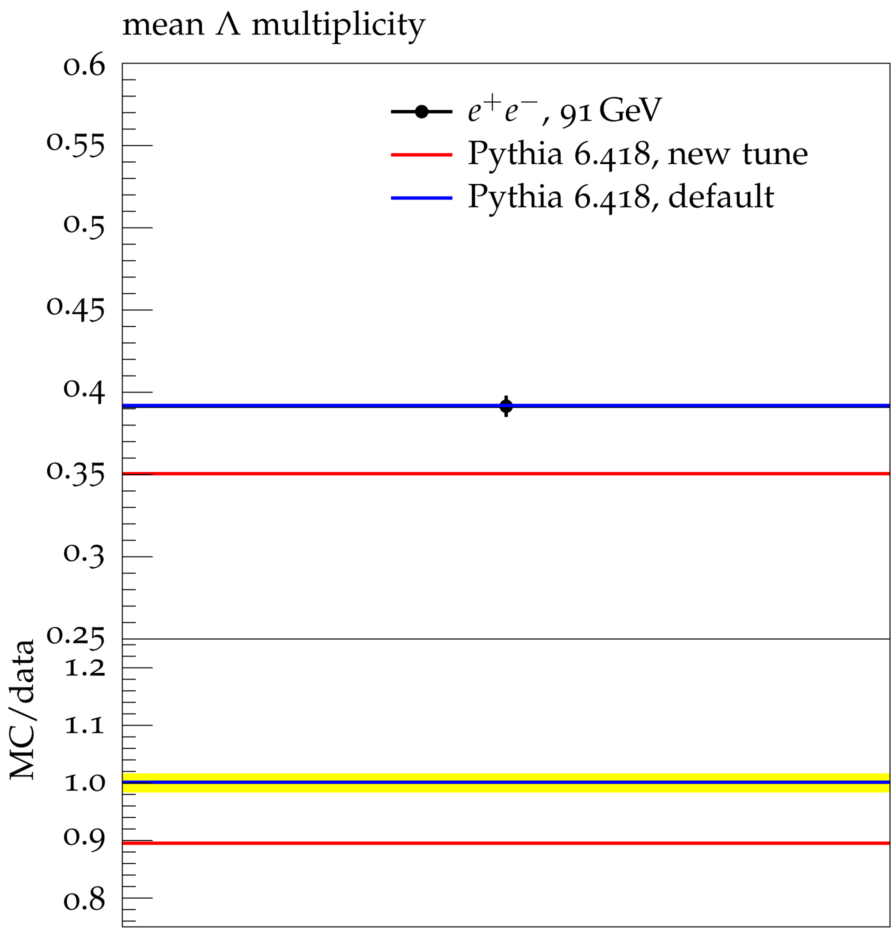
Pythia 6 – New Tunings: LEP



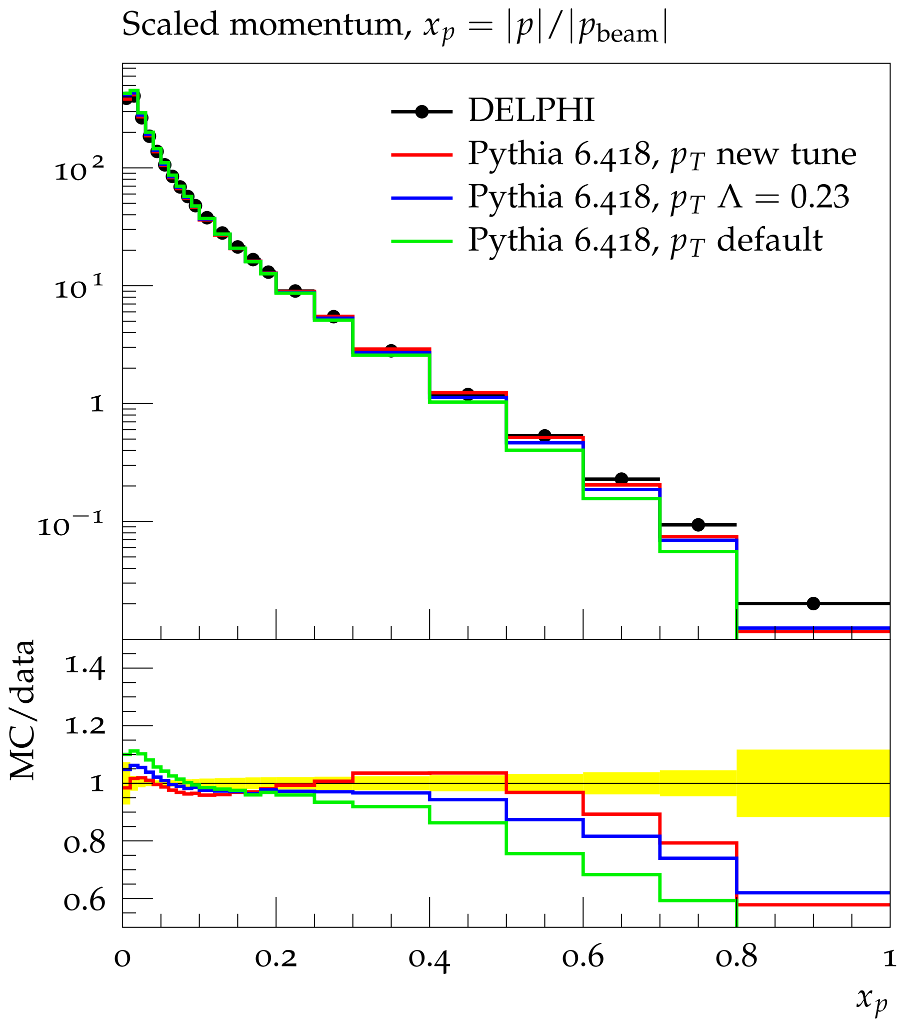


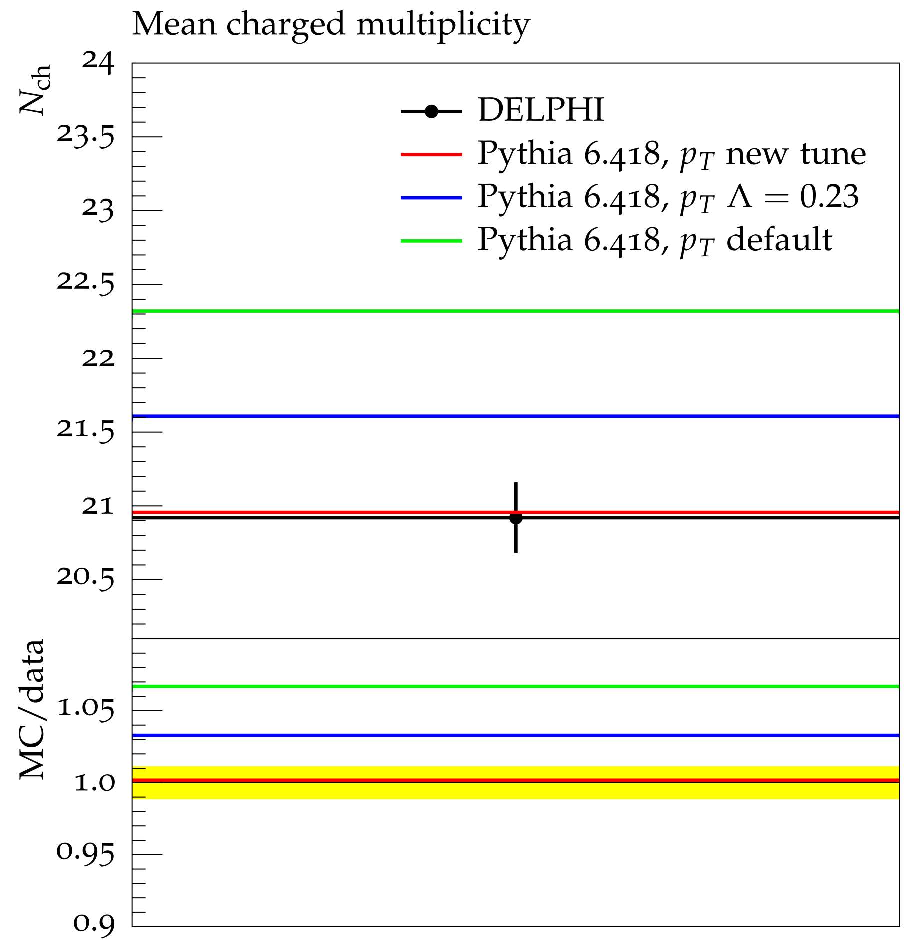
Pythia 6 – New Tunings: LEP



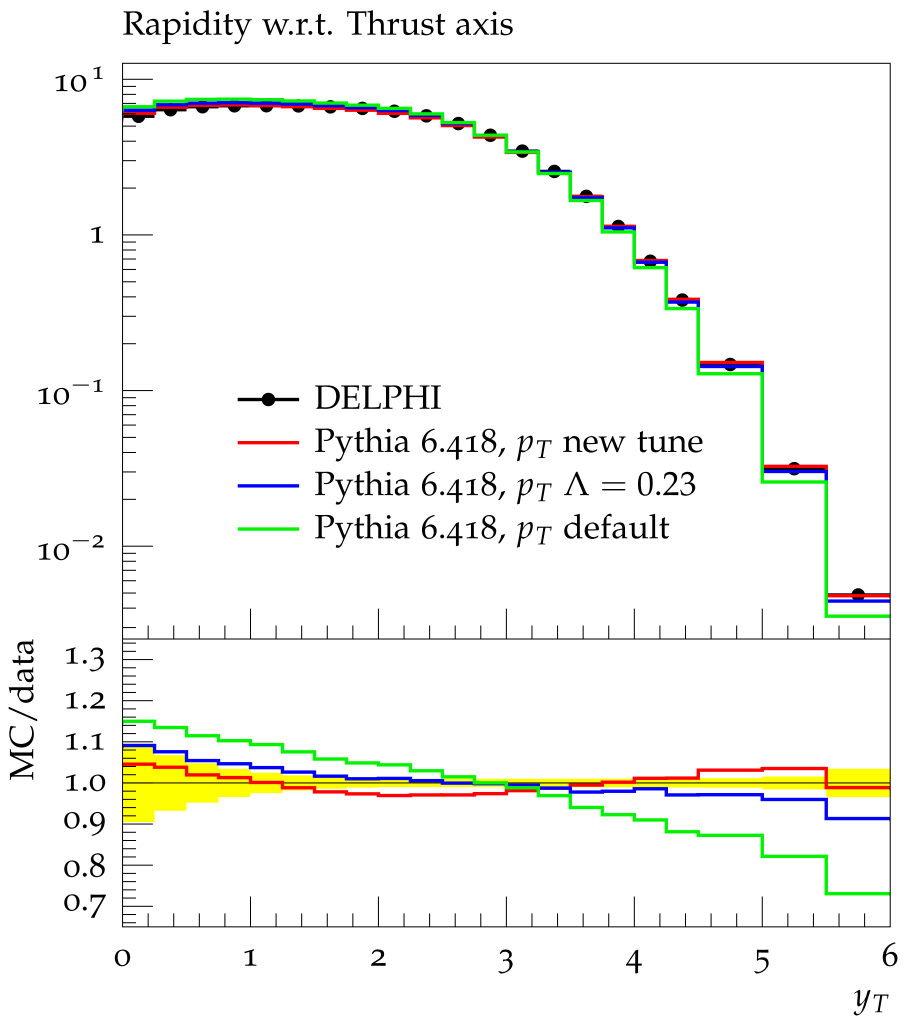


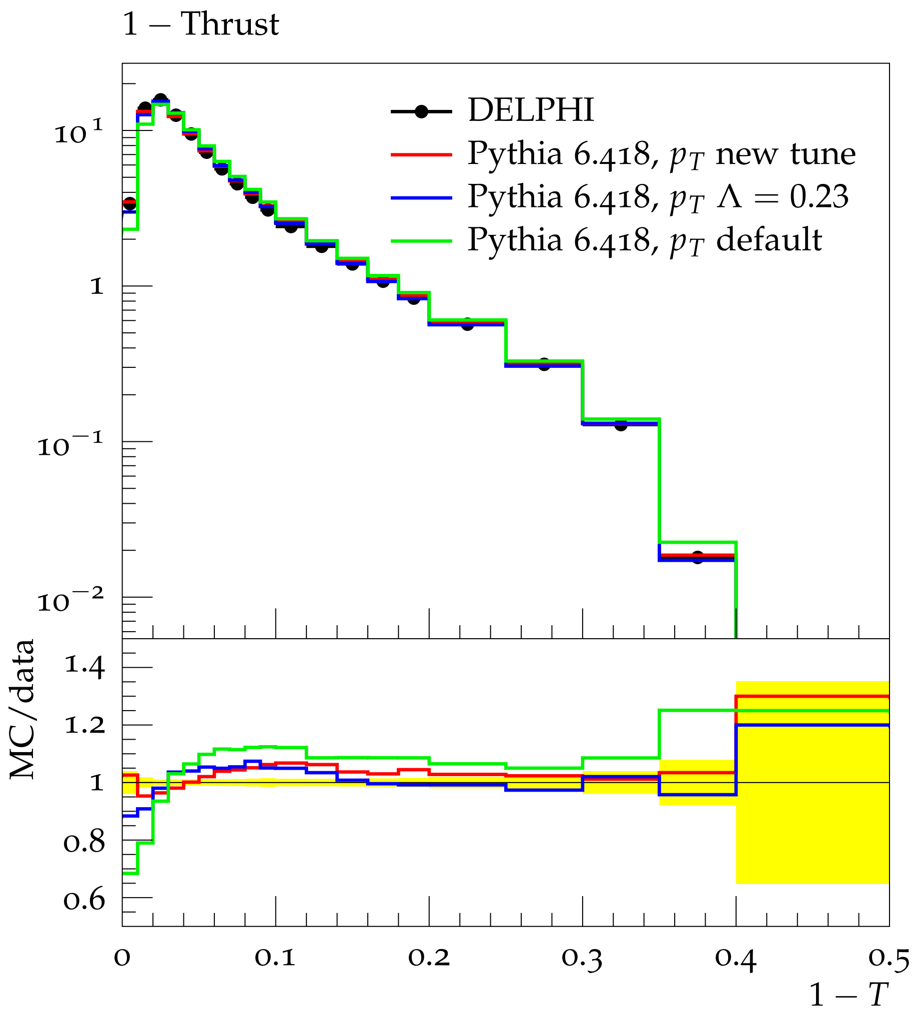
Pythia 6 – New Tunings: LEP



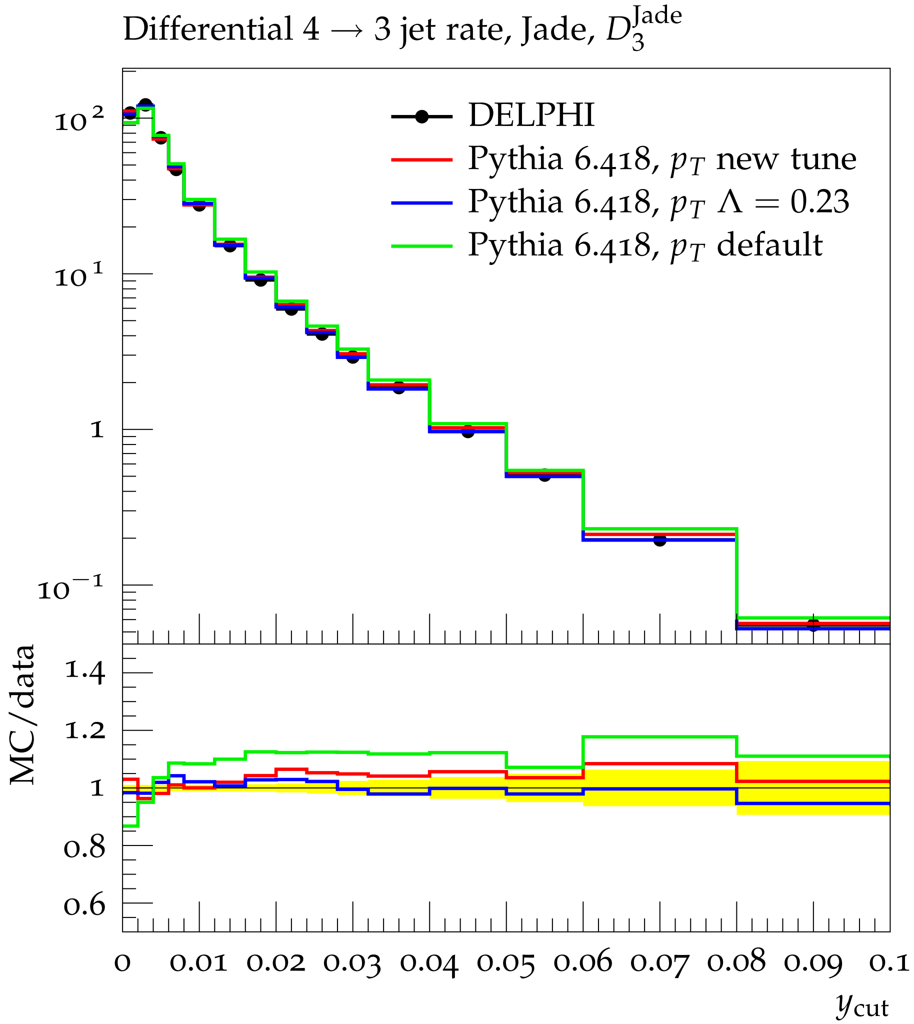


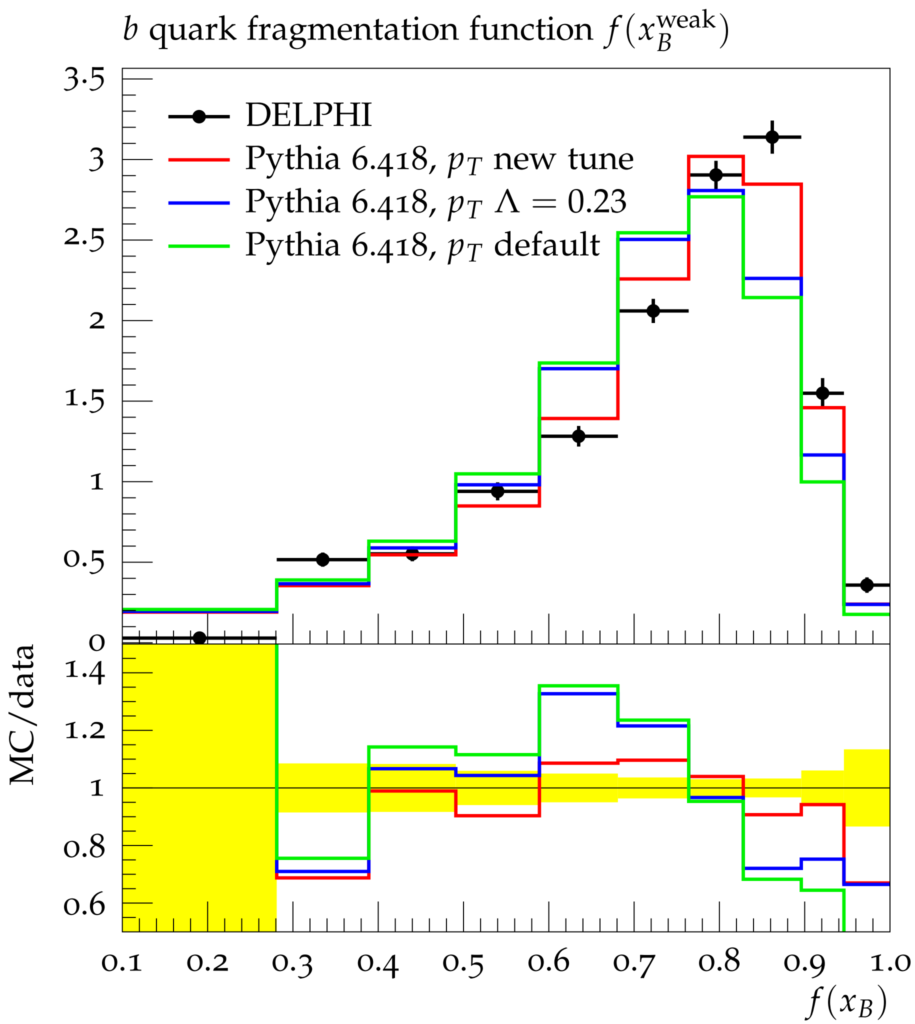
Pythia 6 – New Tunings: LEP





Pythia 6 – New Tunings: LEP



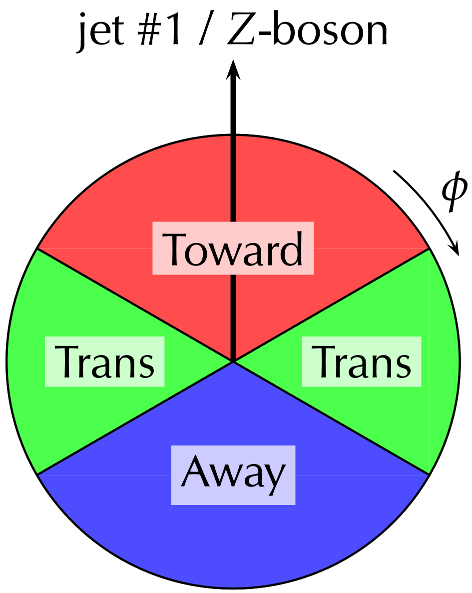


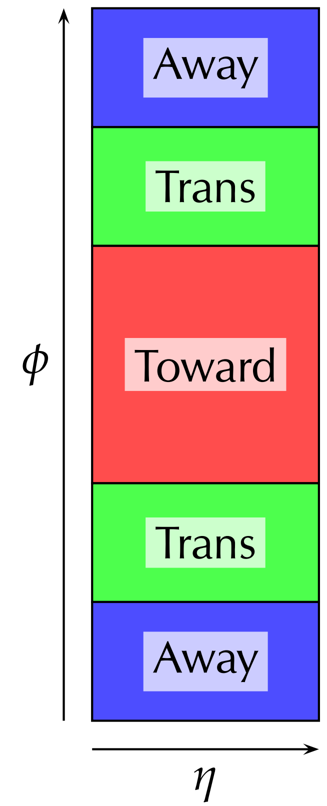
Pythia 6 – New Tunings: Tevatron

Several new UE tunes for Pythia 6 on the market, based on the LEP tunes shown on the last slides (Professor based tunes, manual tunes by Peter Skands, and combinations of Rick Field's tunes with our LEP tunes – all available in 6.420 via PYTUNE).

Using >50 distributions from CDF and DØ:

- CDF N_{ch} at 630 and 1800 GeV
- CDF Run-I Zp_T
- CDF Run-I jets
- CDF Run-II Drell-Yan
- CDF Run-II leading jet
- CDF Run-II $\langle p_T \rangle$ vs N_{ch}
- DØ Run-II jet correlations





Challenges for LHC Predictions

- Scaling behaviour of the IR regularization scale:

  The current Pythia default (0.16) for the power of the rescaling term is clearly too small. But from 630 and 1960 GeV it's a loooong stretch to 14 TeV.

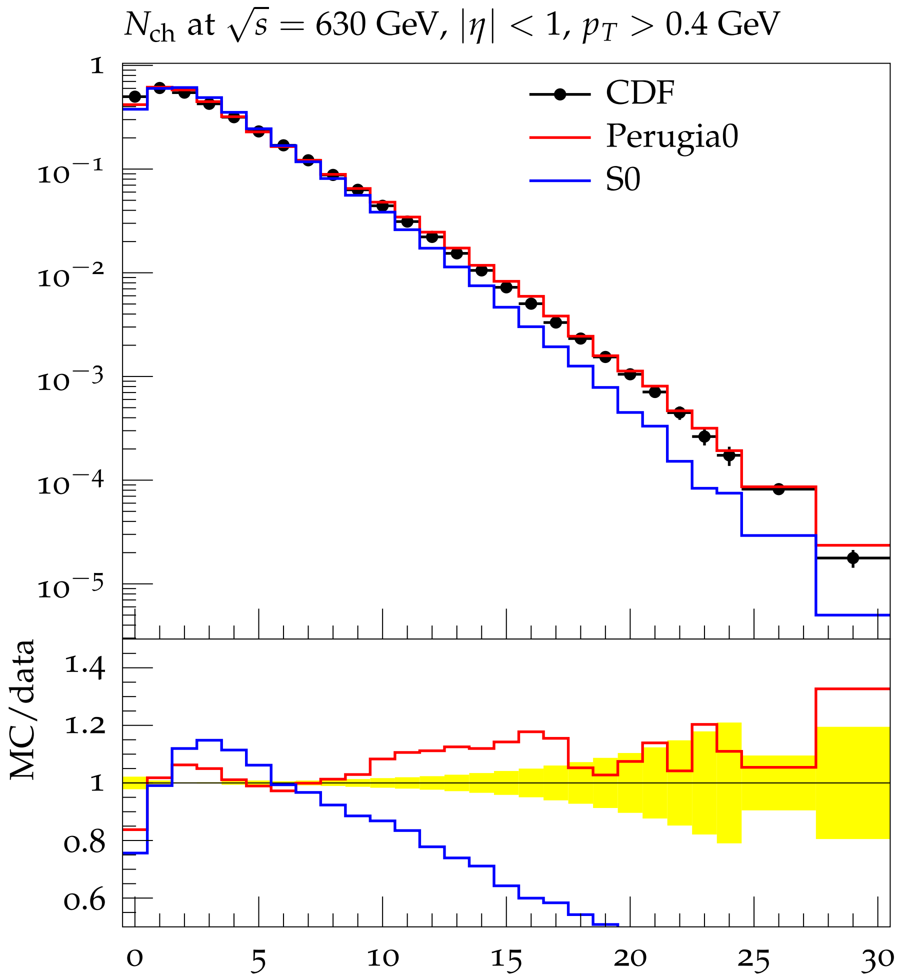
  I want to include 200 GeV RHIC pp-data as soon as possible!

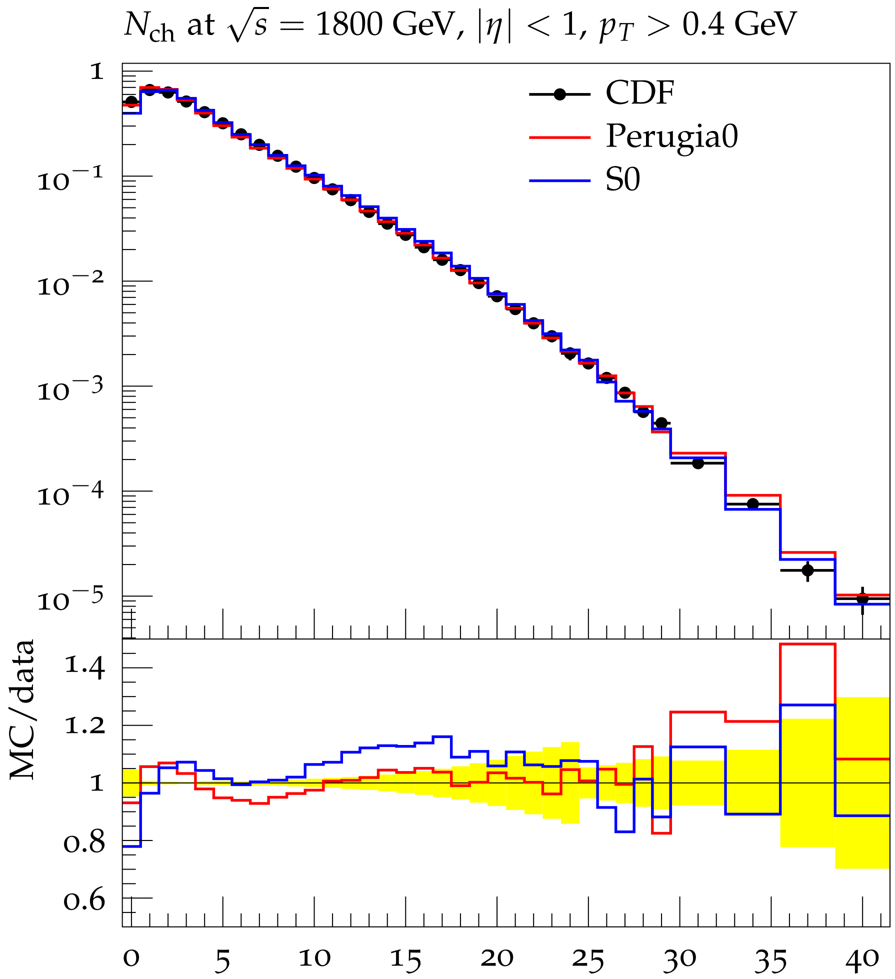
- PDF parameterizations:

  The low-x region will be important at the LHC, but uncertainties there are relatively large.

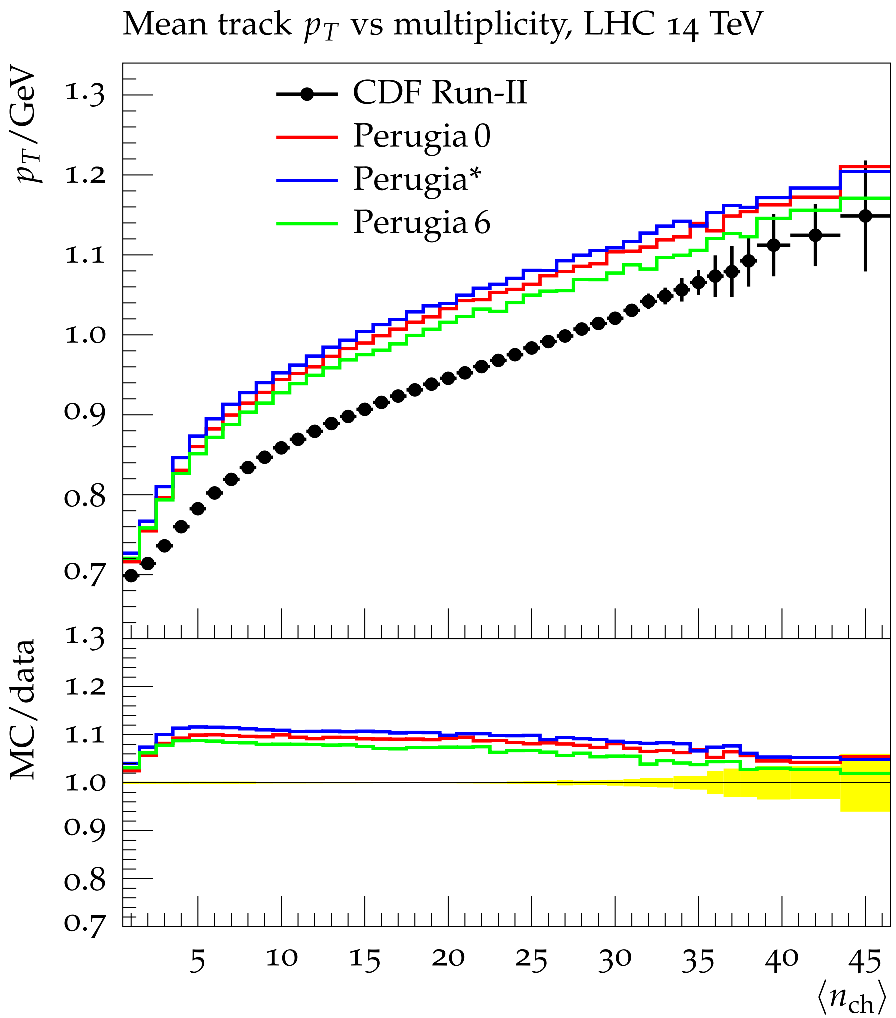
$\Rightarrow$ The quality of our predictions really depends on the data we have available, and we'll need early LHC data to retune the generators.

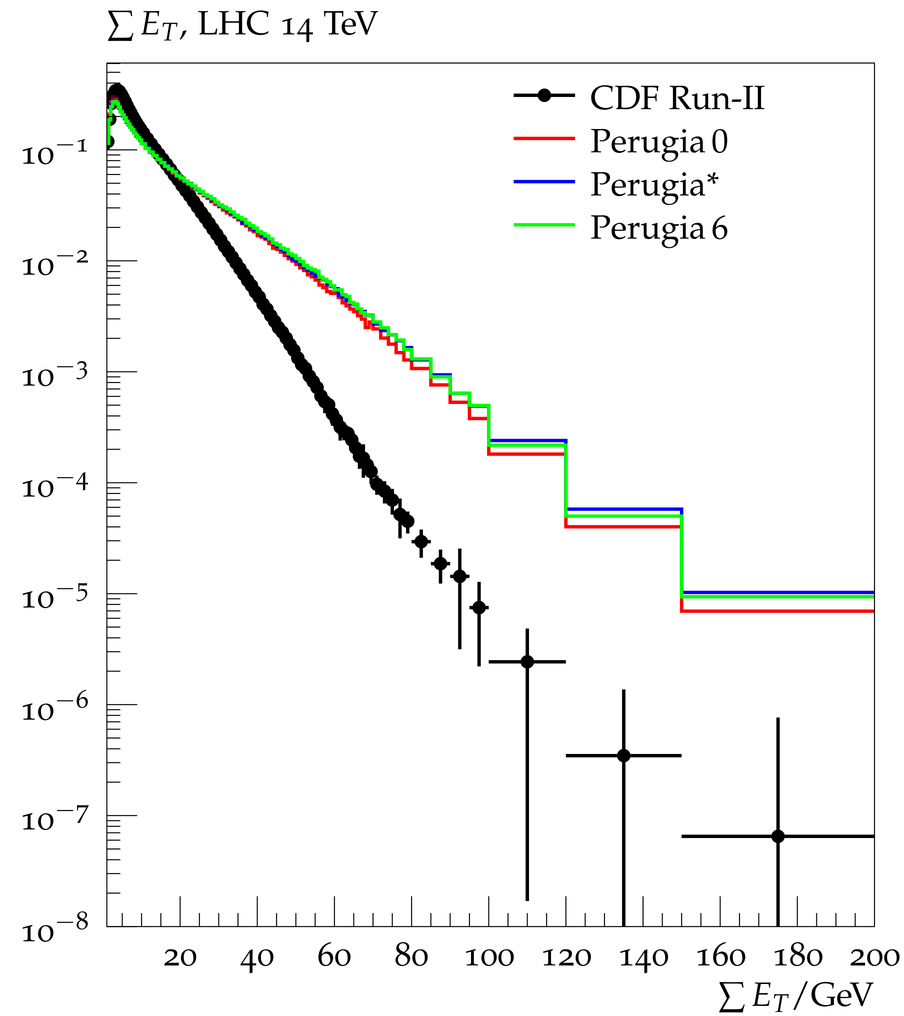
Pythia 6 – Tevatron: energy scaling



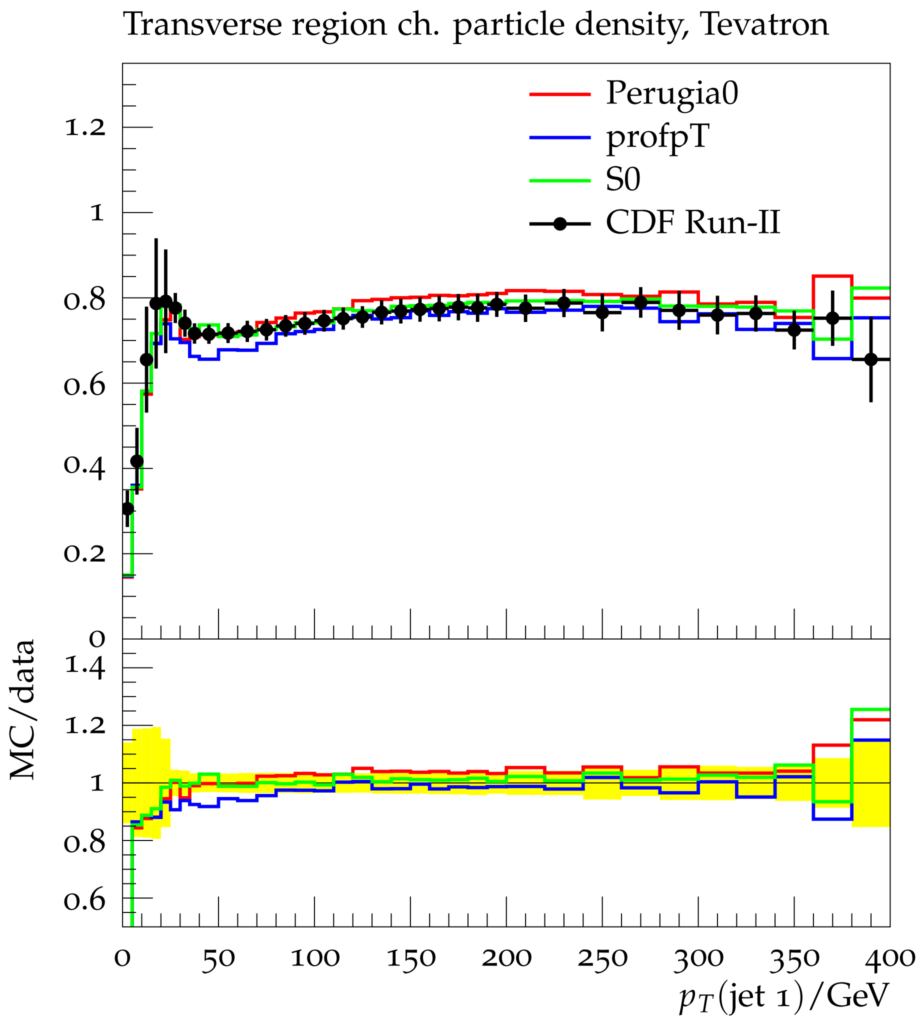


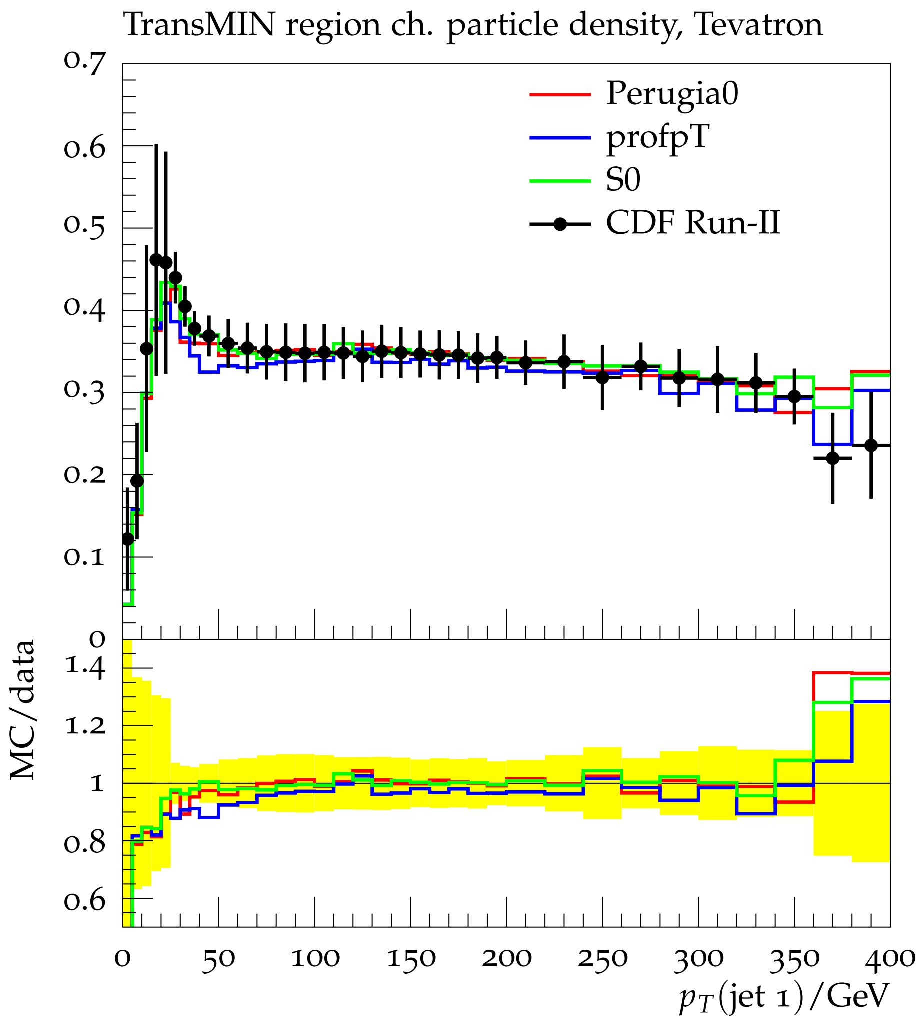
Pythia 6 – LHC: CTEQ5L vs LO* vs CTEQ6L1



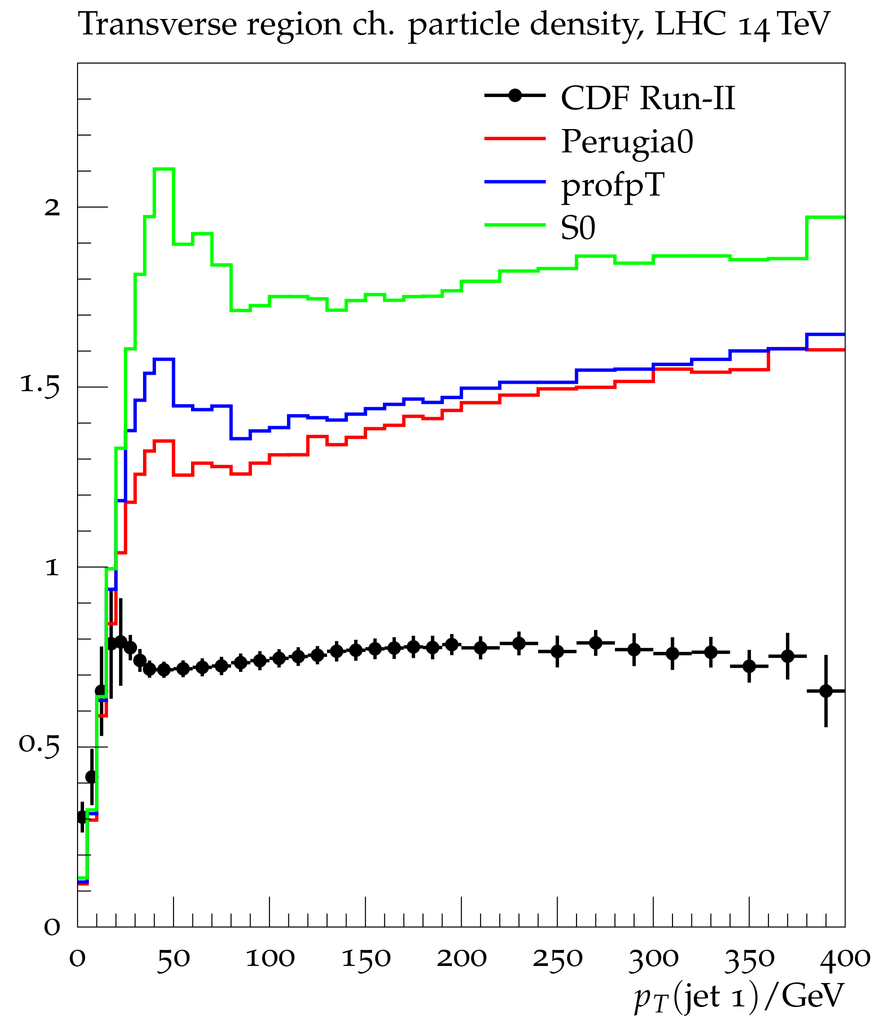


Pythia 6 – Tevatron

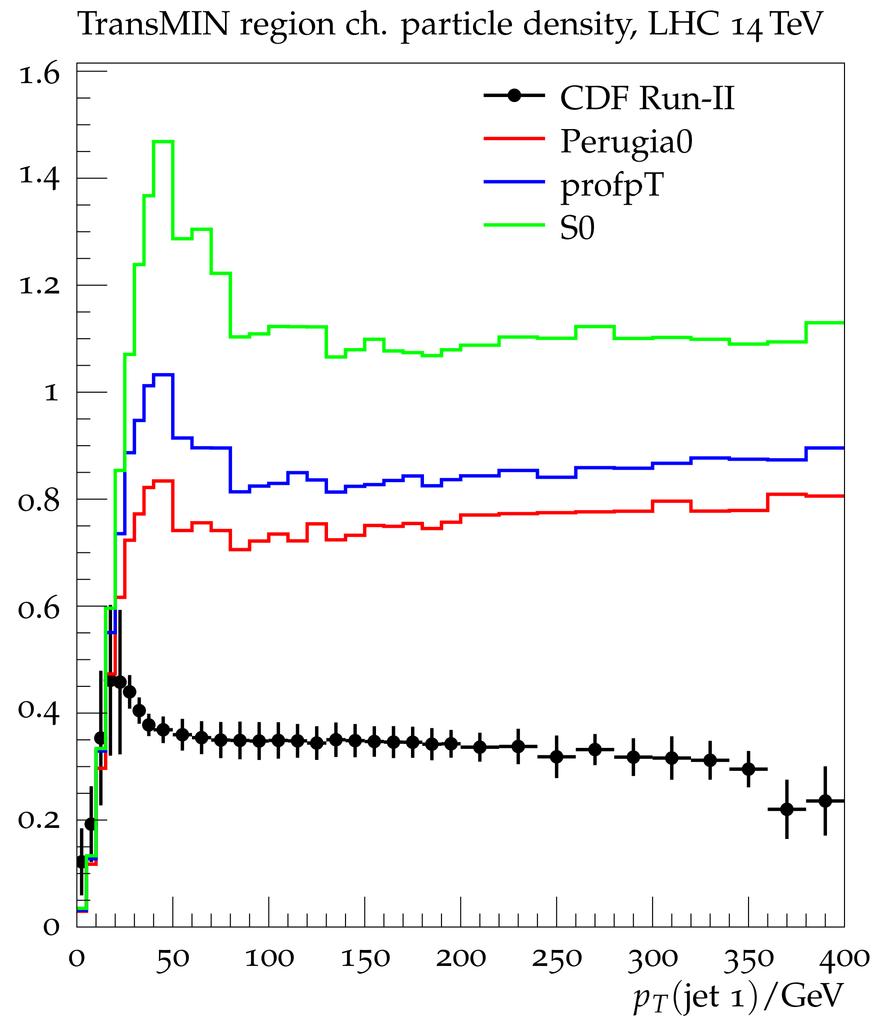




Pythia 6 – LHC

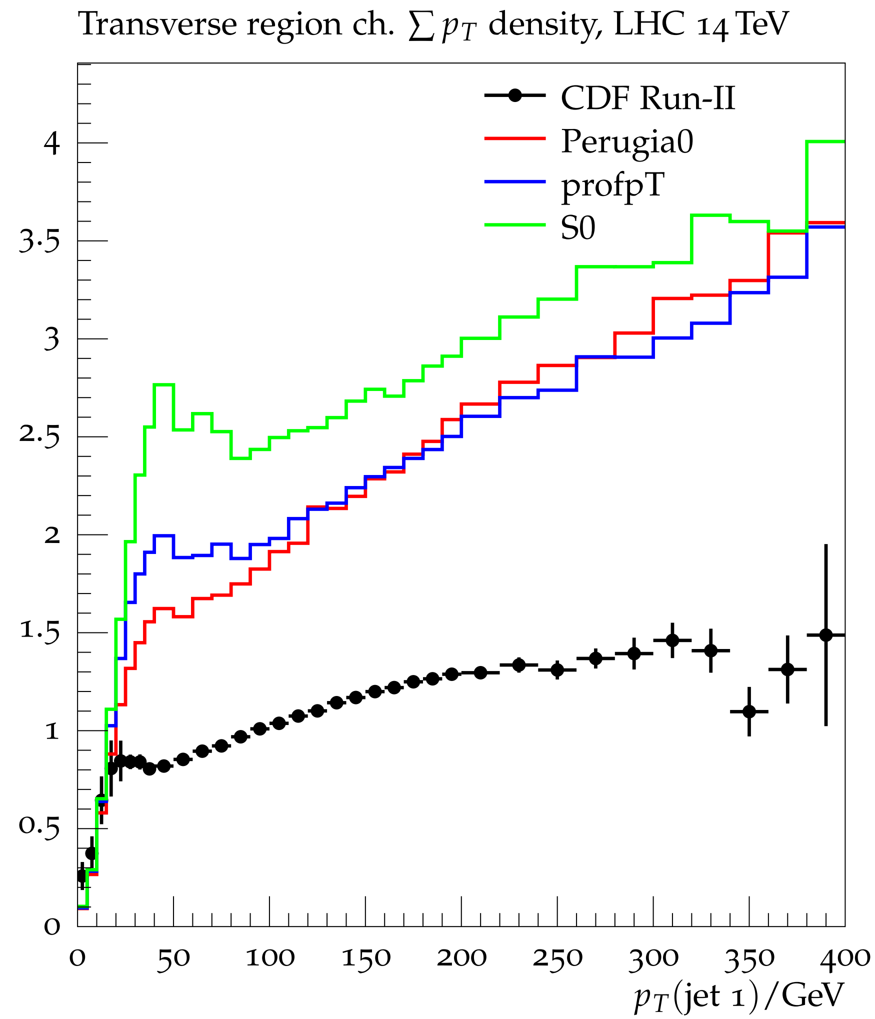


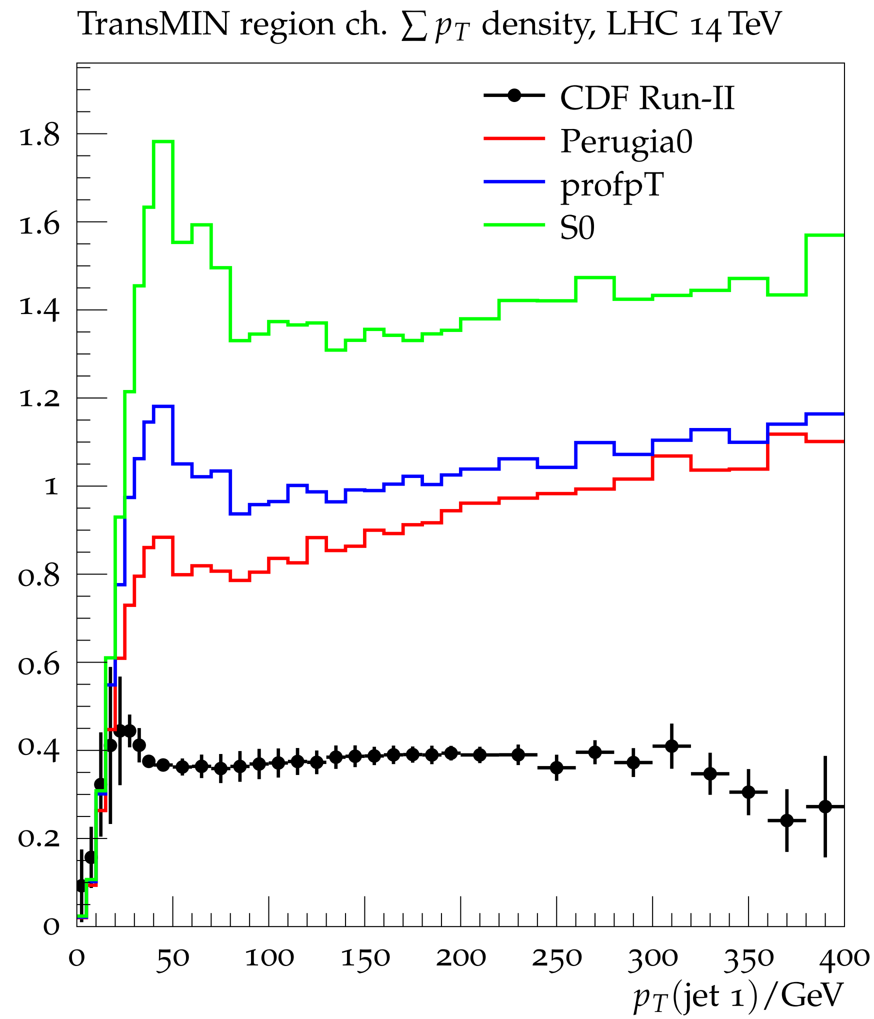
Transverse region ch. particle density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
0
0.5
1
1.5
2
0
50
100
150
200
250
300
350
400
p_T(jet 1)/GeV



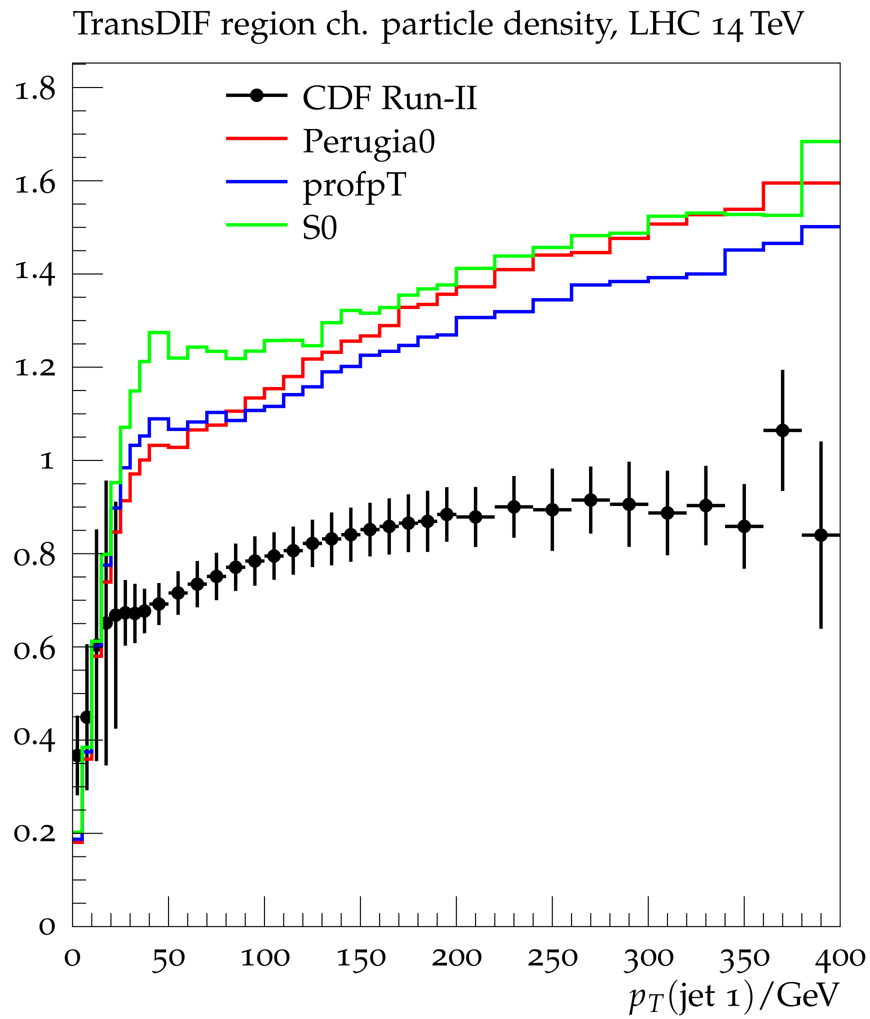
TransMIN region ch. particle density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
0
50
100
150
200
250
300
350
400
p_T(jet 1)/GeV

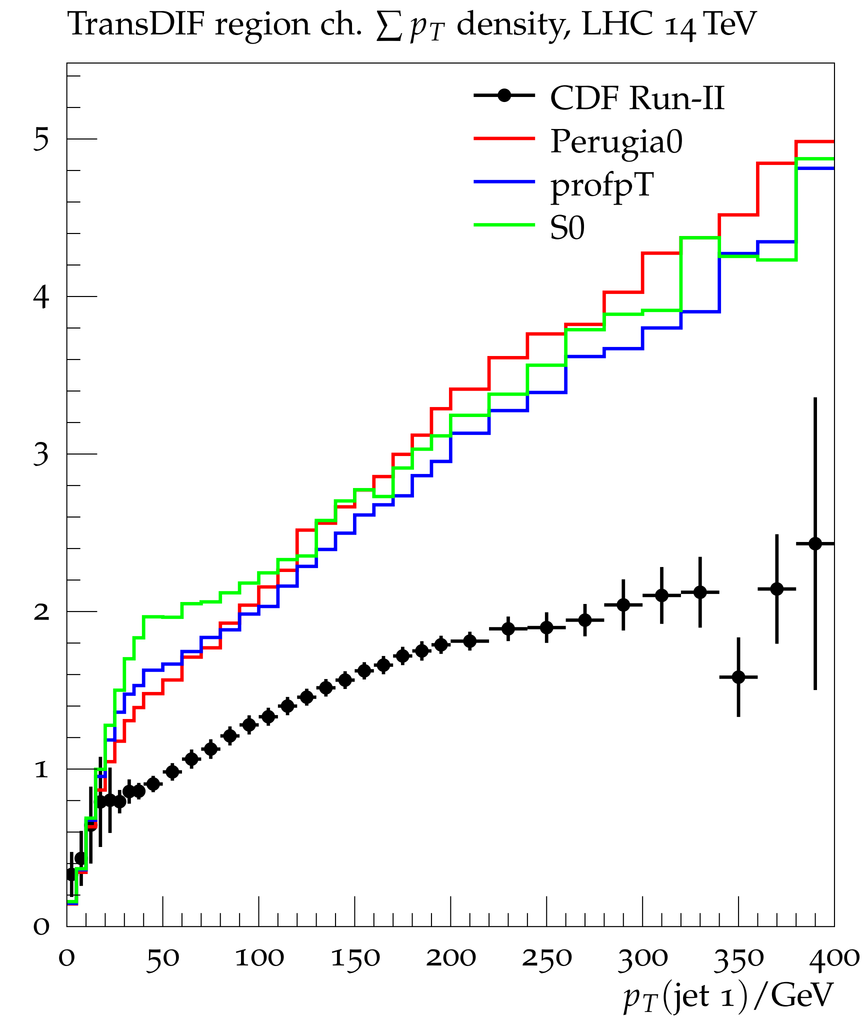
Pythia 6 – LHC


Transverse region ch. $\sum p_T$ density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
0
0.5
1
1.5
2
2.5
3
3.5
4
0
50
100
150
200
250
300
350
400
p_T(jet 1)/GeV


TransMIN region ch. $\sum p_T$ density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
0
50
100
150
200
250
300
350
400
p_T(jet 1)/GeV

Pythia 6 – LHC


TransDIF region ch. particle density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
p_T(jet 1)/GeV


TransDIF region ch. $\sum p_T$ density, LHC 14 TeV
CDF Run-II
Perugia0
profpT
S0
p_T(jet 1)/GeV

Outlook

Things on our list for the (near) future:

- Include more data in the tuning and validation, especially RHIC
- Tune Pythia 8 (on its way, hopefully done in May)
- Tune Sherpa: shower, hadronization (on its way, but very slow)
- Tune Herwig++ (on its way, but slow)
- Provide side-by-side comparisons of main tunings and generators

Summary

Monte Carlo tuning is needed for improvement of data description, and for understanding and developing models.

We have tools for systematic tuning and validation in place, and we strive for global improvements of the models.

Last year nobody really knew what to expect at the LHC.
Now we can make predictions – let's shrink the uncertainties more and place the bets!