Tuning and Generator Comparison

Hendrik Hoeth
(Lund University)

Overview

Motivation	–	“why” and “what’s the problem”
Strategy	–	“how to tune”
Tunings	–	“plots, plots, plots”
Outlook	–	“where we are” and “where to go next”

Motivation

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

The models have free parameters which are a priori unknown: flavour ratios, q_0^2, Lund A and B, intrinsic k_T, ...

We want the MC to describe the data the best possible way. So the parameters need to be tuned.

Even parameters like α_s need to be optimised!

Problems

The 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.

A 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:

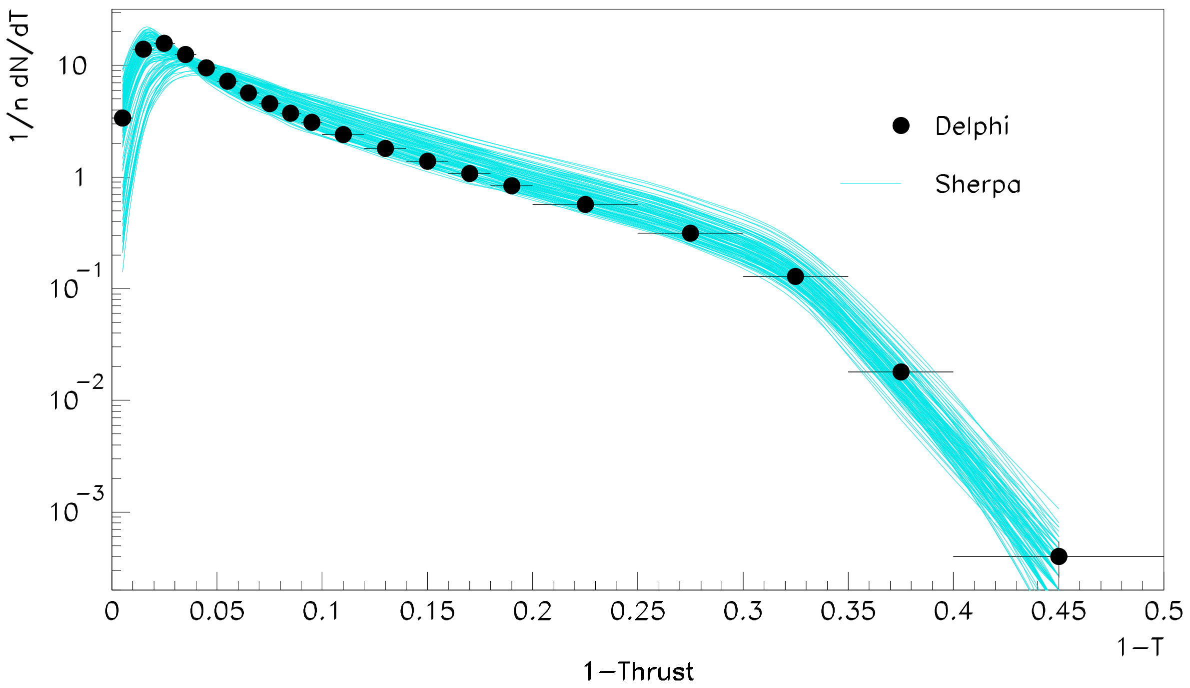
- Tune everything that is important.
- But remember: Each additional parameter adds one dimension to the parameter space.

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:

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

Include all the relevant data distributions in the fit!

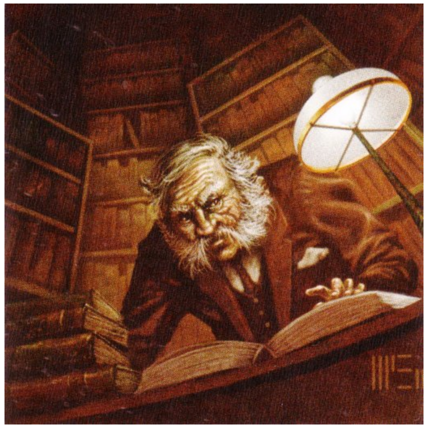
This fit only takes seconds (as compared to days or weeks for a brute force approach).

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.

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:

- 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

Flavour parameters:

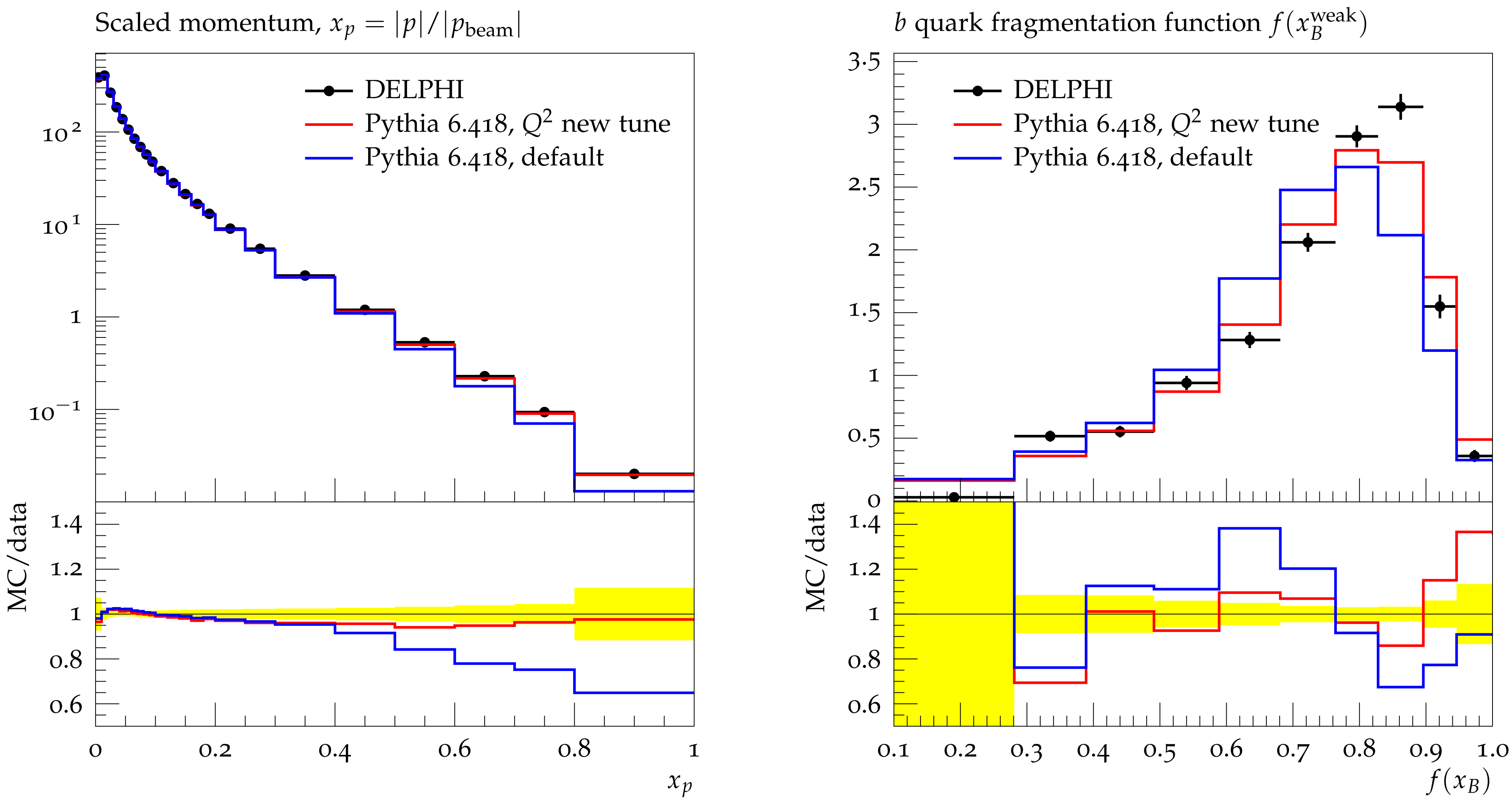
	default	tuned	
PARJ(1)	0.1	0.073	di-quark suppression
PARJ(2)	0.3	0.2	strange suppression
PARJ(3)	0.4	0.94	strange di-quark suppression
PARJ(4)	0.05	0.032	spin-1 di-quark suppression
PARJ(11)	0.5	0.31	spin-1 light meson
PARJ(12)	0.6	0.4	spin-1 strange meson
PARJ(13)	0.75	0.54	spin-1 heavy meson
PARJ(25)	1.0	0.63	η suppression
PARJ(26)	0.4	0.12	η' suppression

Pythia 6 – New Tunings: LEP

Fragmentation parameters:

	default	Q^2 shower	p_T shower	
MSTJ(11)	4	5	5	Lund-Bowler frag.
PARJ(21)	0.36	0.325	0.313	σ_q
PARJ(41)	0.3	0.5	0.49	a
PARJ(42)	0.58	0.6	1.2	b
PARJ(47)	1.0	0.67	1.0	r_b
PARJ(81)	0.29	0.29	0.257	Λ
PARJ(82)	1.0	1.65	0.8	PS cut-off

Pythia 6 – New Tunings: LEP



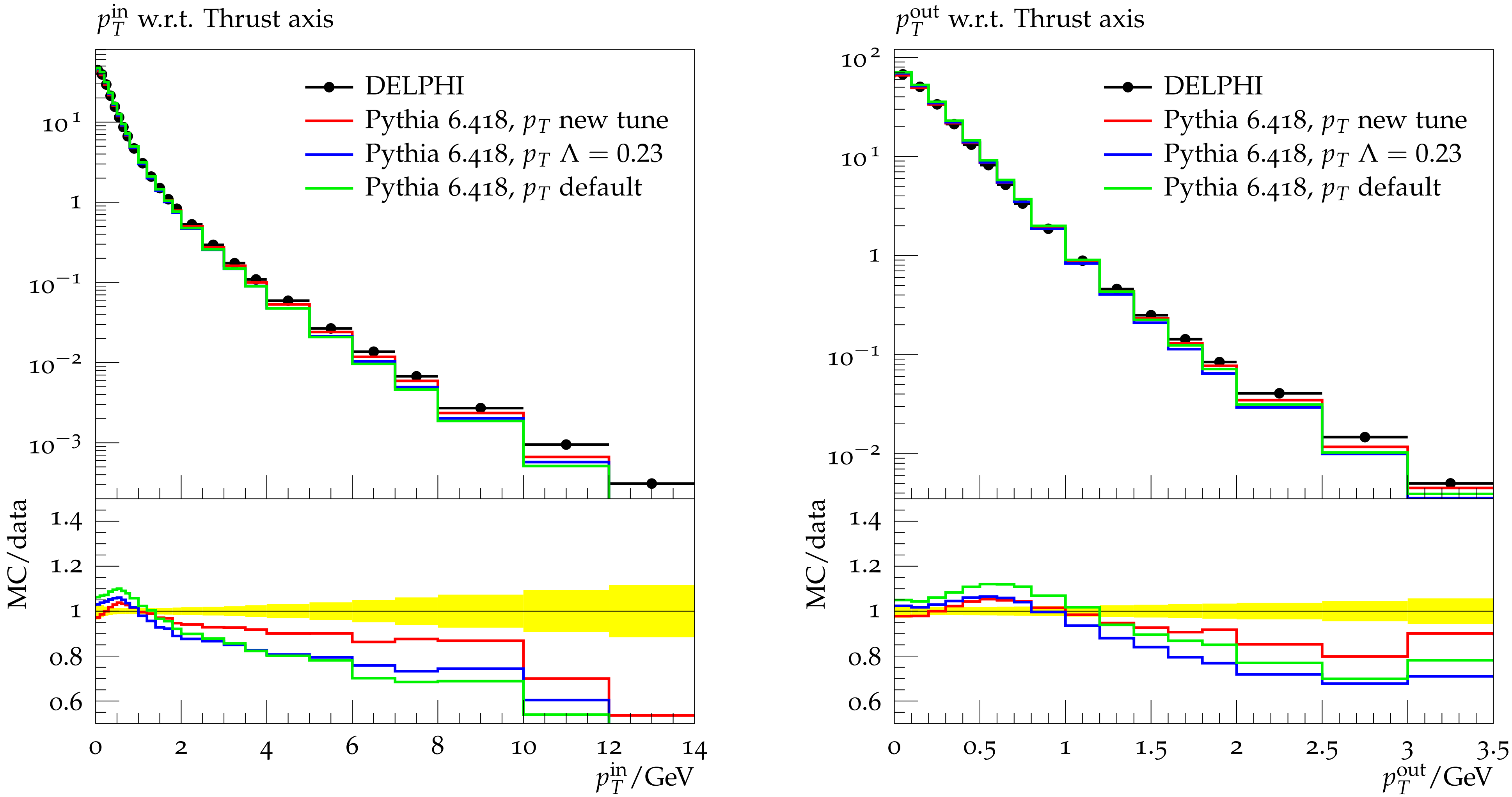
Pythia 6 – New Tunings: LEP

The next slides show the p_T ordered shower. Keep in mind:

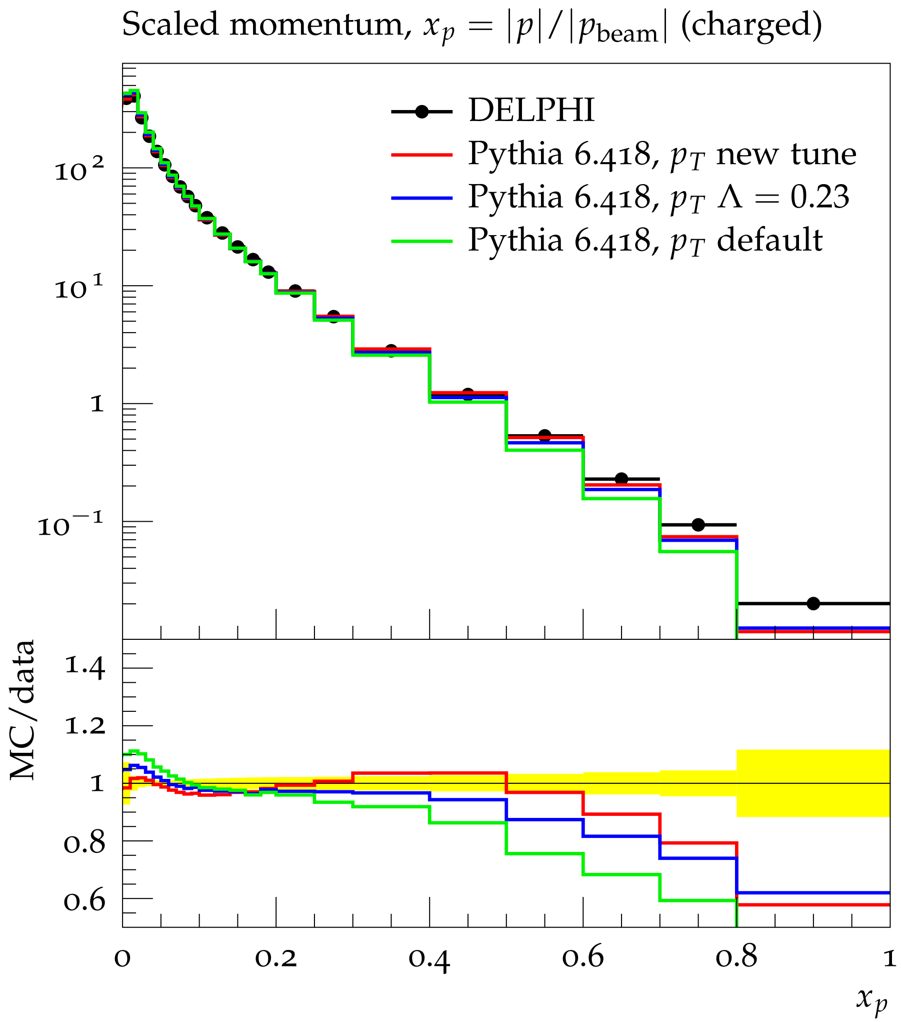
- DØ and CDF use tune S0, equivalent to the "$\Lambda = 0.23$" curve.
- ATLAS uses Arthur Moraes' tune, equivalent to the "default" curve.

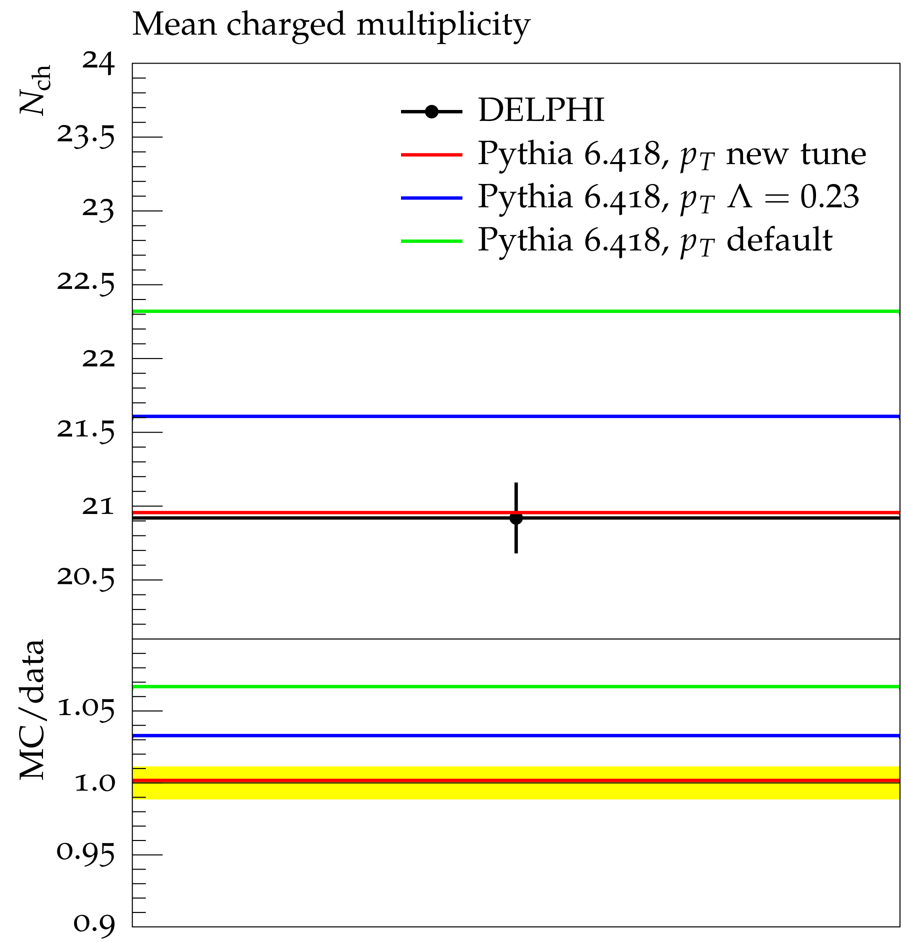
The differences you see *do* affect Tevatron/LHC distributions.
Just turning on the p_T ordered shower is *not* sufficient!

Pythia 6 – New Tunings: LEP

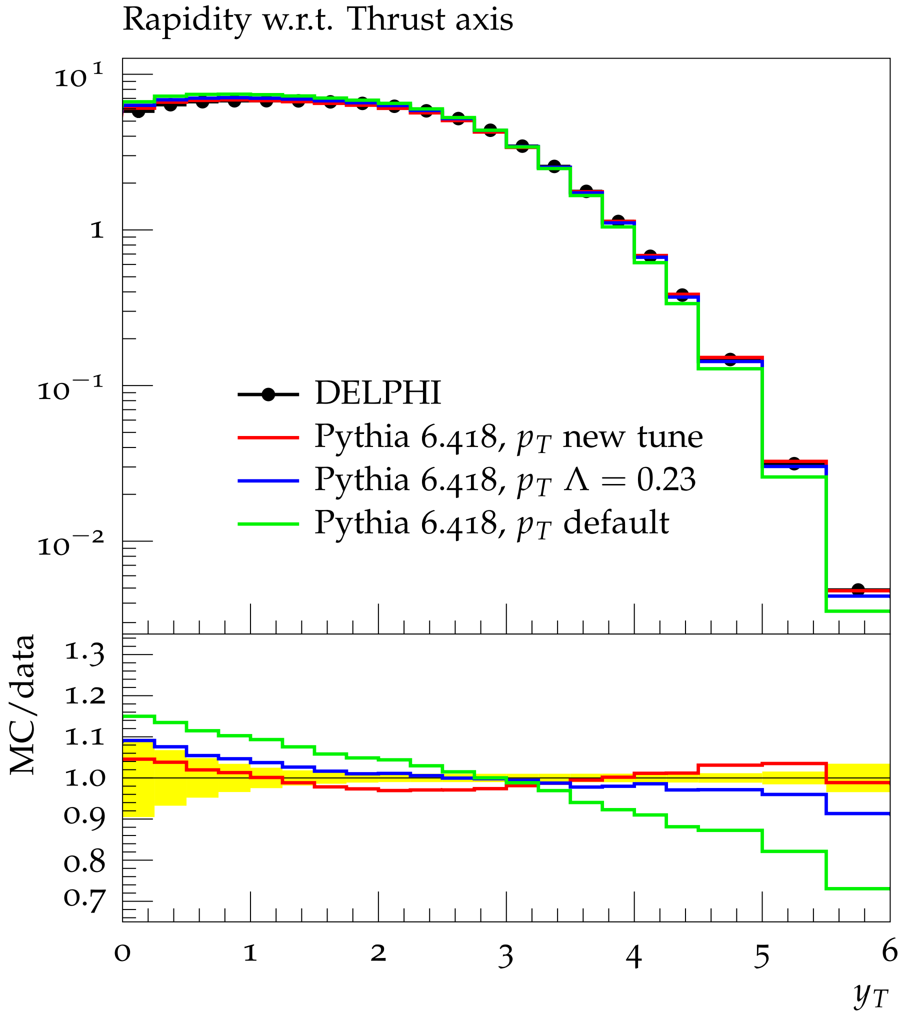


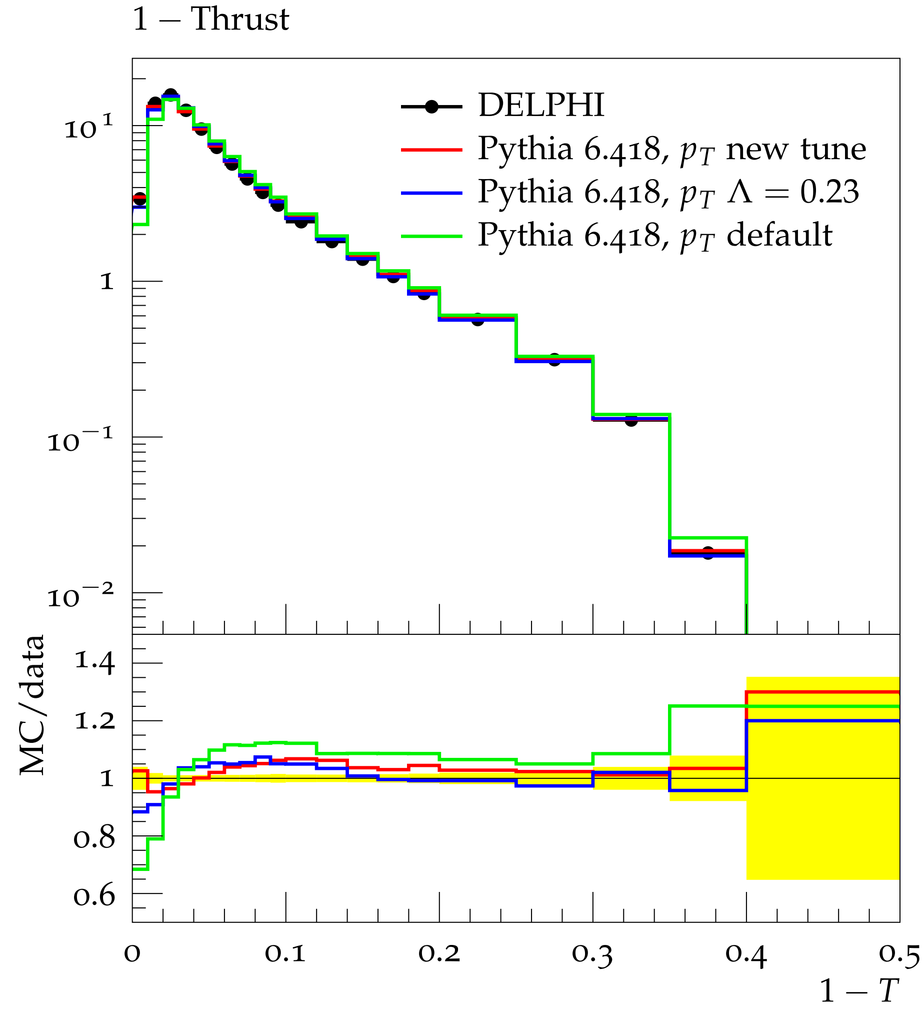
Pythia 6 – New Tunings: LEP



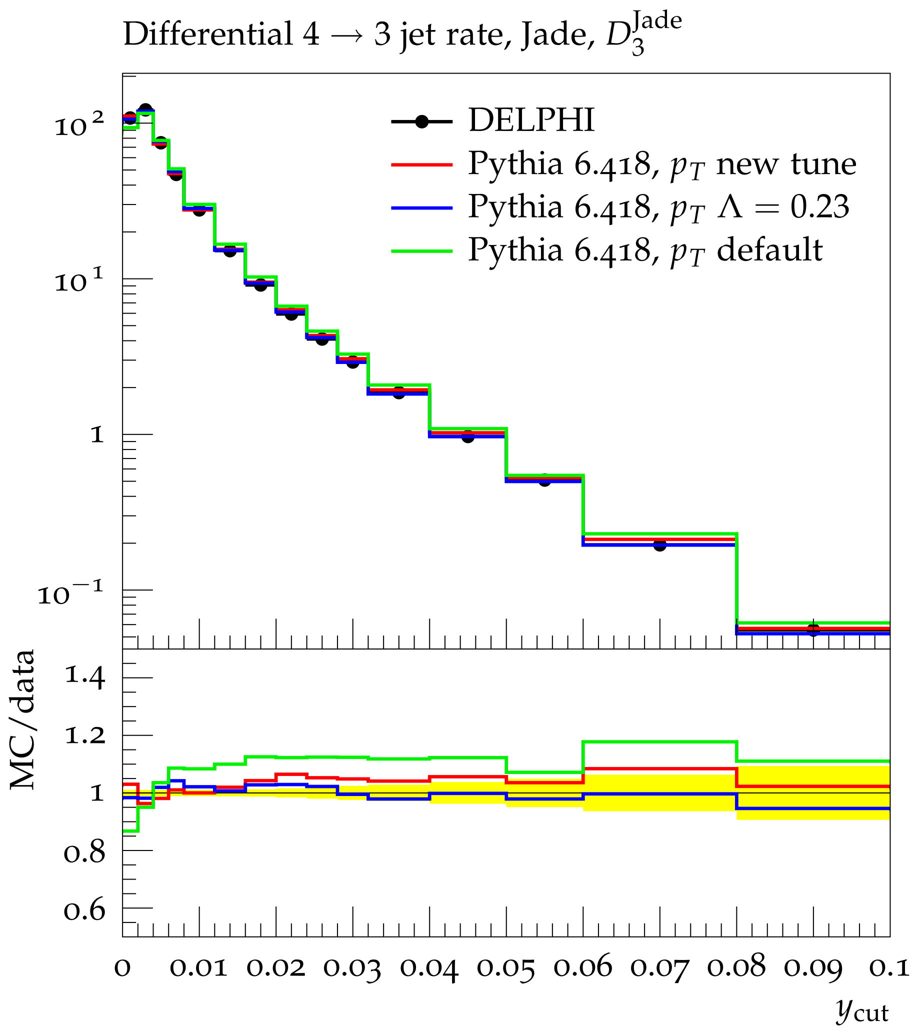


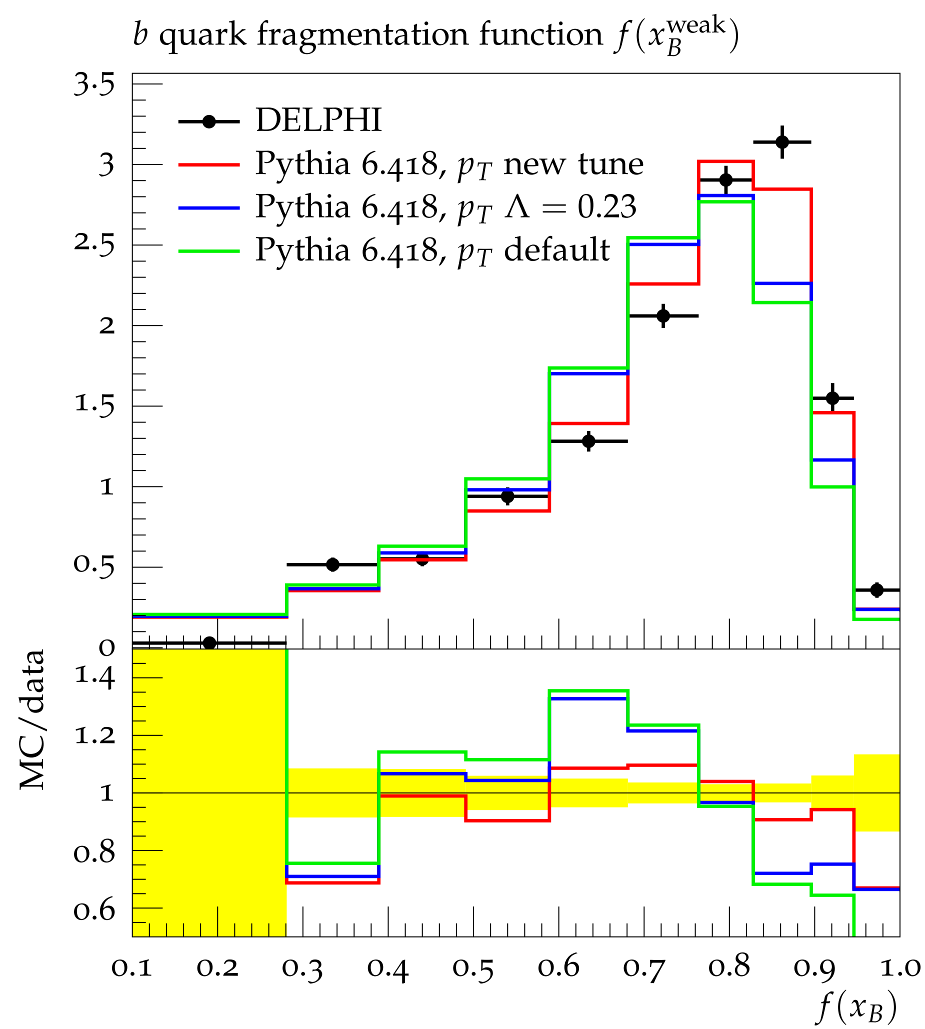
Pythia 6 – New Tunings: LEP





Pythia 6 – New Tunings: LEP



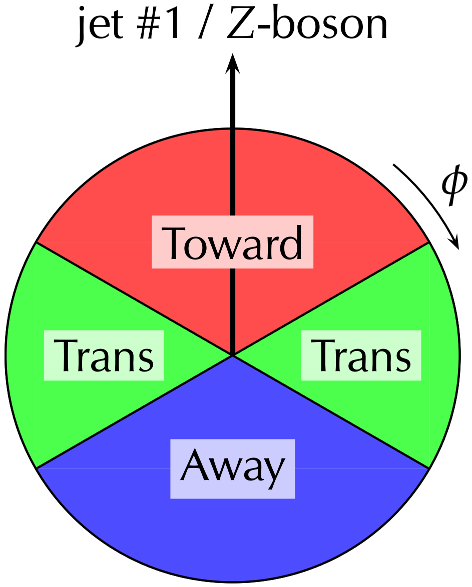


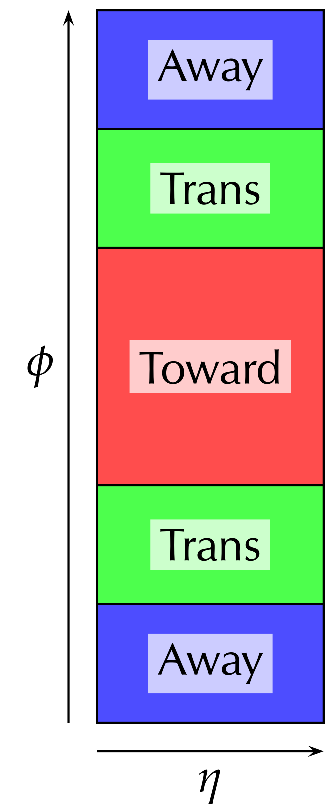
Pythia 6 – New Tuning: Tevatron

We have a new UE tune for Pythia 6 with Q^2 ordered shower and old MPI model, based on the LEP tune shown on the last slides.

Using > 50 distributions from CDF and DØ:

- CDF Run-I $Z p_T$
- 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





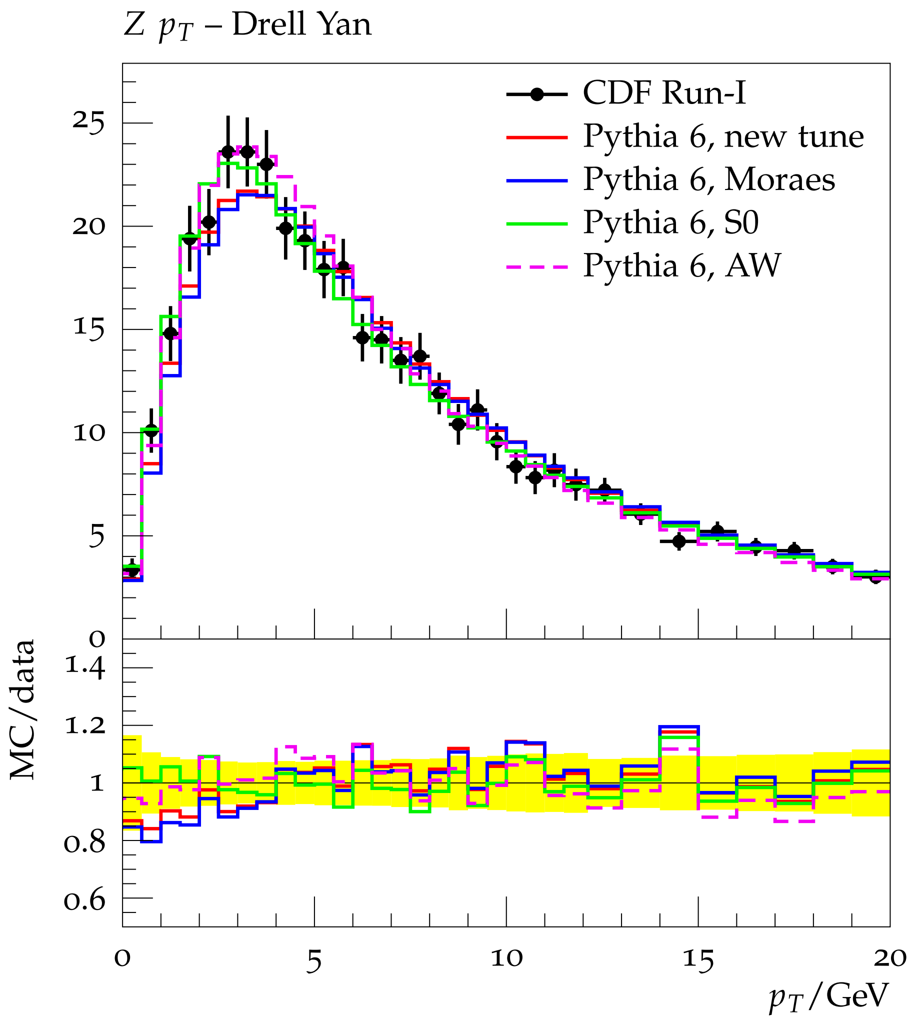
NB: A UE tune with the p_T ordered shower and the new MPI model is in progress.

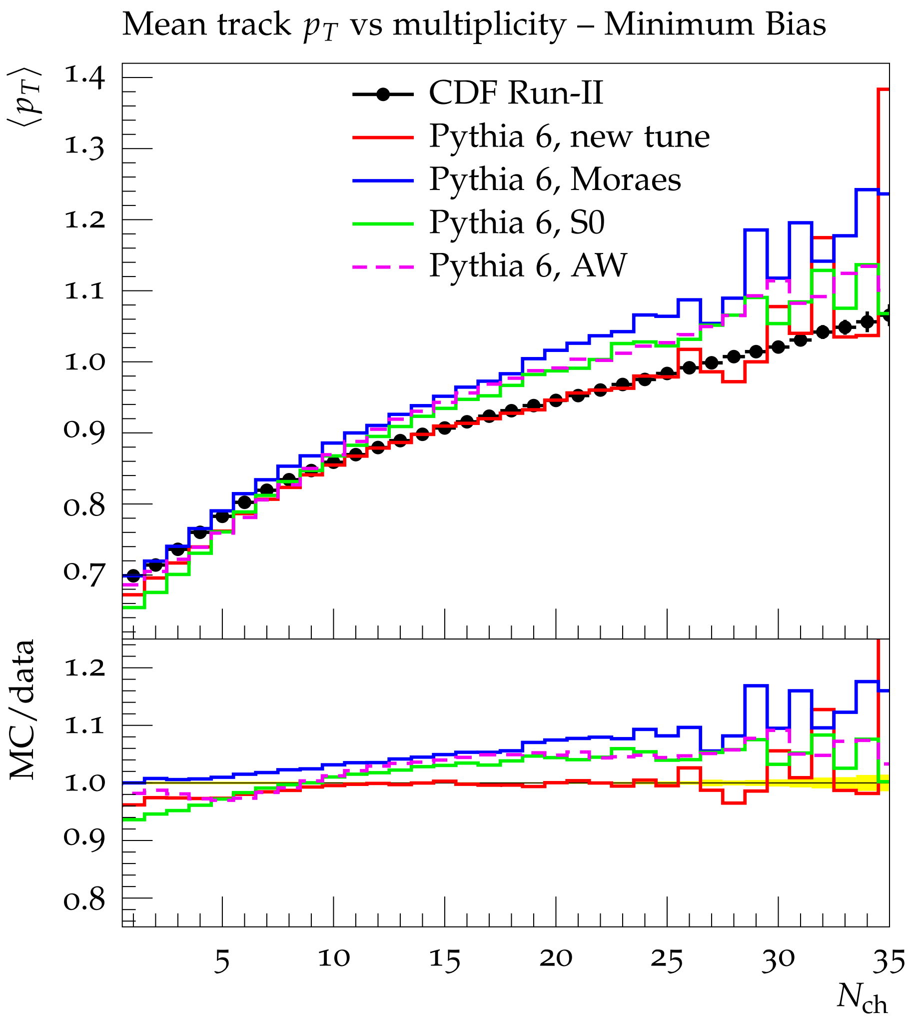
Pythia 6 – New Tuning: Tevatron

Tune uses LEP tuned parameters as shown before, Q^2 shower:

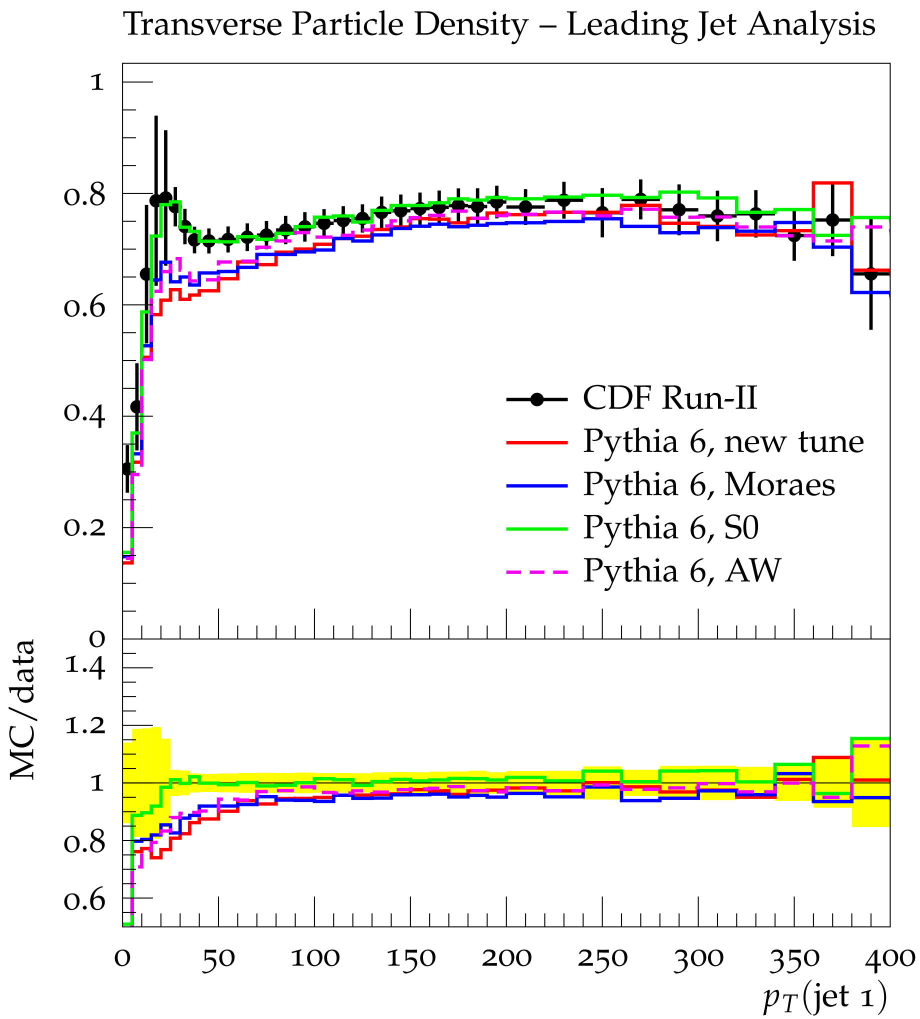
	default	tune DW	new tune	
PARP(62)	1.0	1.25	2.97	ISR cut-off
PARP(64)	1.0	0.2	0.12	ISR scale factor for α_s
PARP(67)	4.0	2.5	2.74	max. virtuality
PARP(82)	2.0	1.9	2.1	p_{T0}
PARP(83)	0.5	0.5	0.84	matter distribution
PARP(84)	0.4	0.4	0.5	matter distribution
PARP(85)	0.9	1.0	0.82	colour connection
PARP(86)	0.95	1.0	0.91	colour connection
PARP(90)	0.16	0.25	0.17	p_{T0} energy evolution
PARP(91)	2.0	2.1	2.0	intrinsic k_T
PARP(93)	5.0	15.0	5.0	intrinsic k_T cut-off

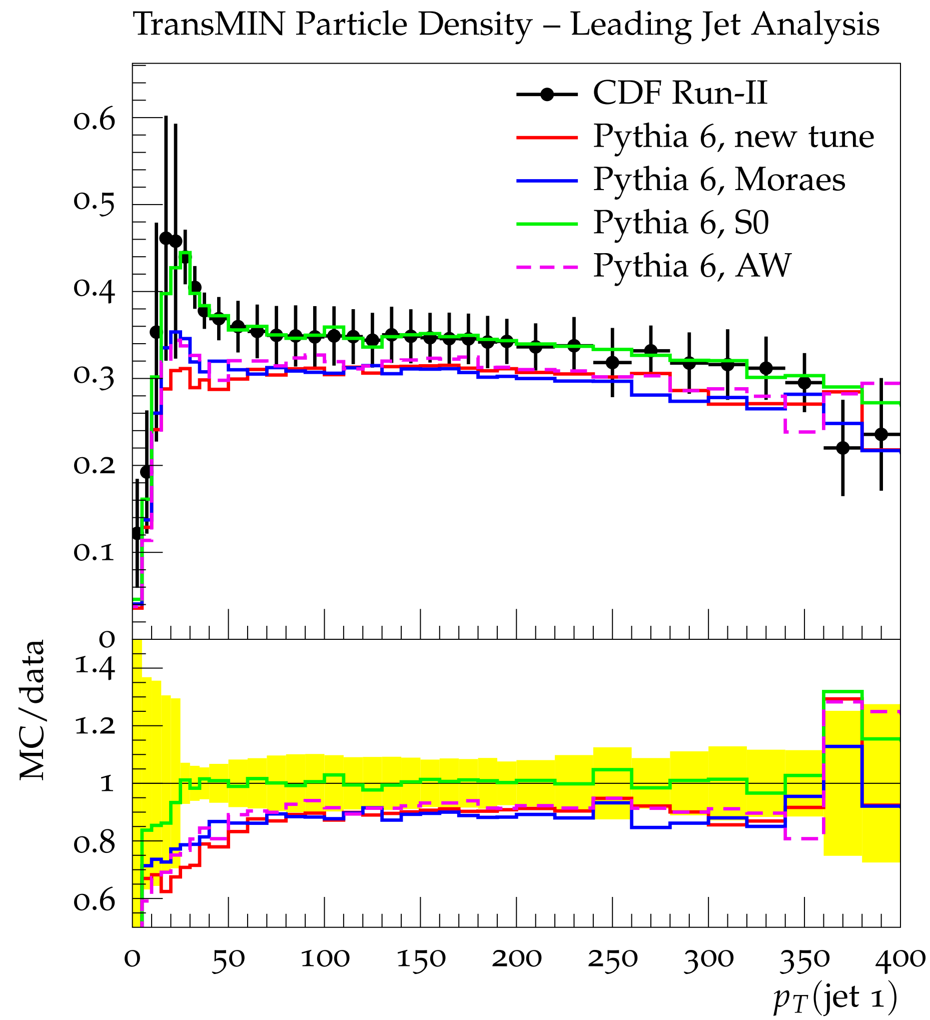
Pythia 6 – Tevatron Comparisons



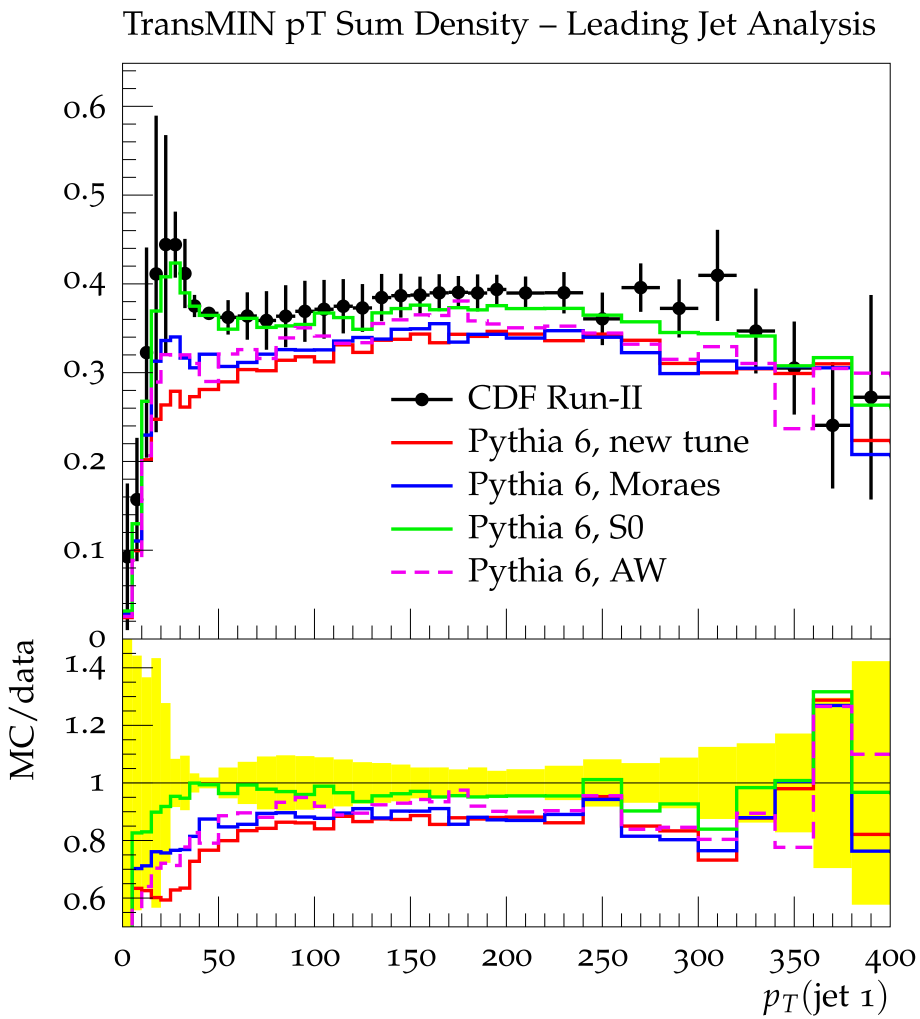


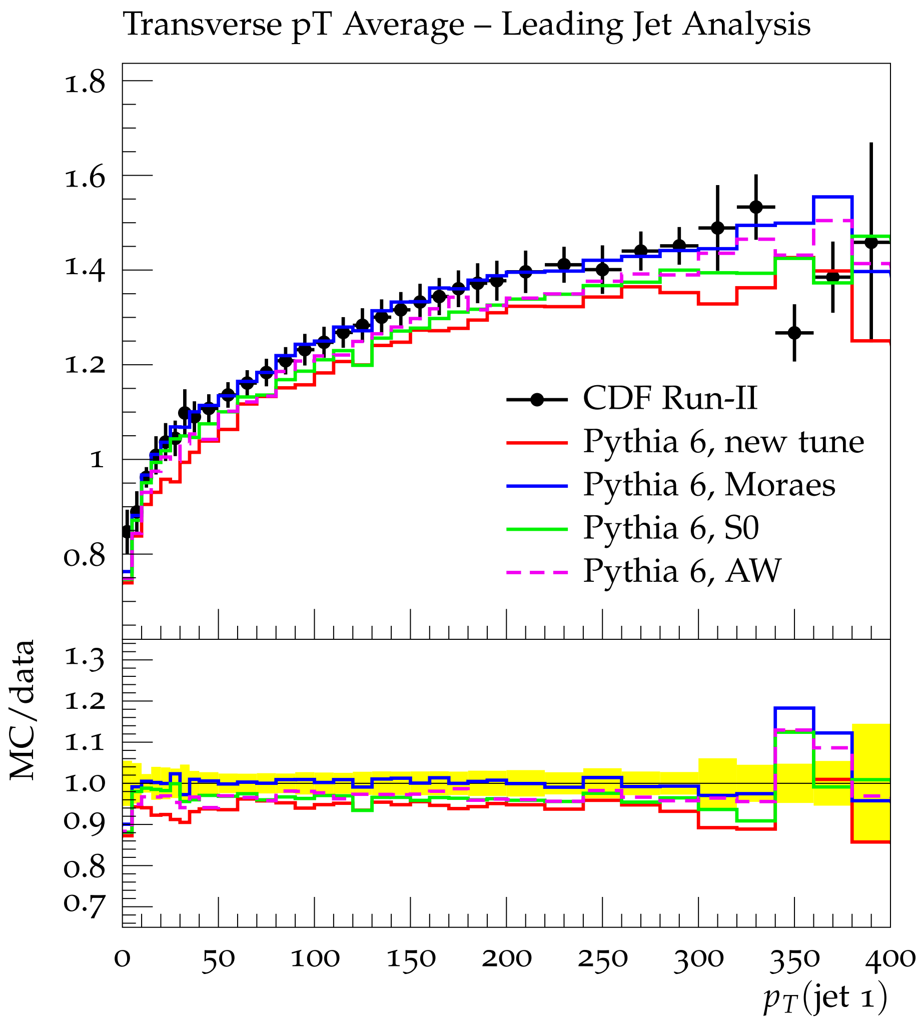
Pythia 6 – Tevatron Comparisons



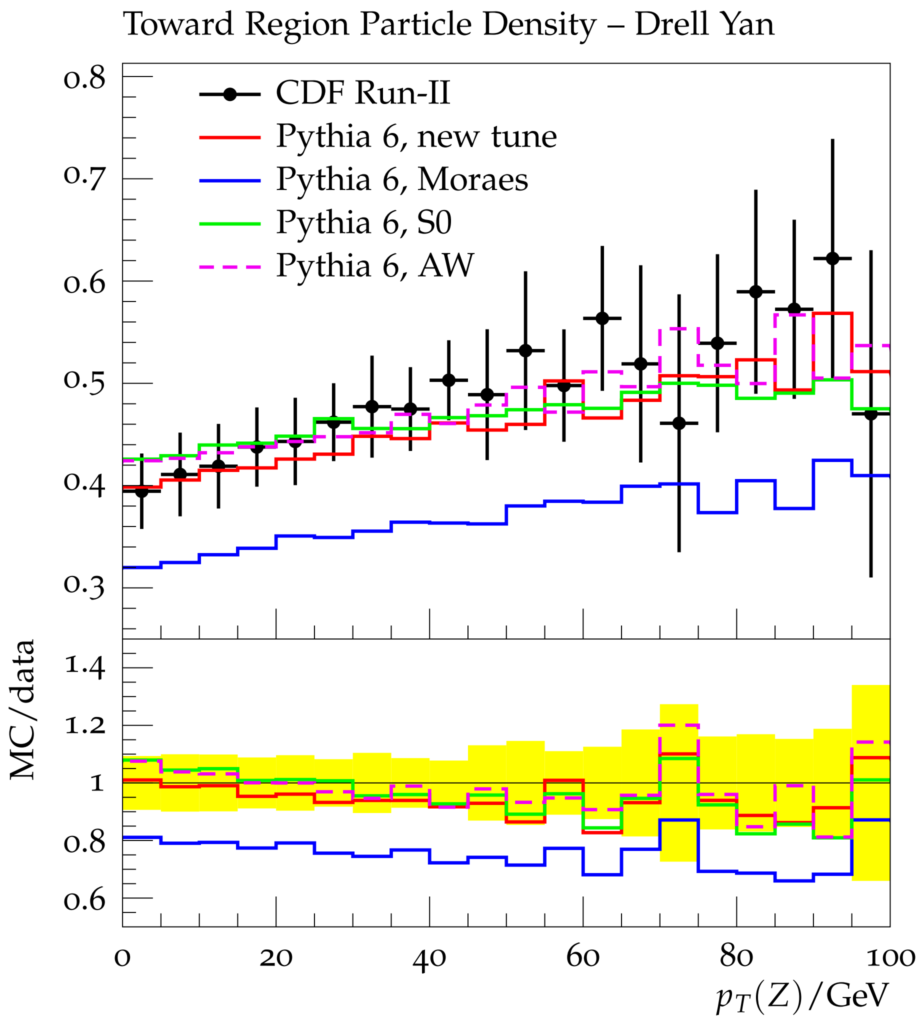


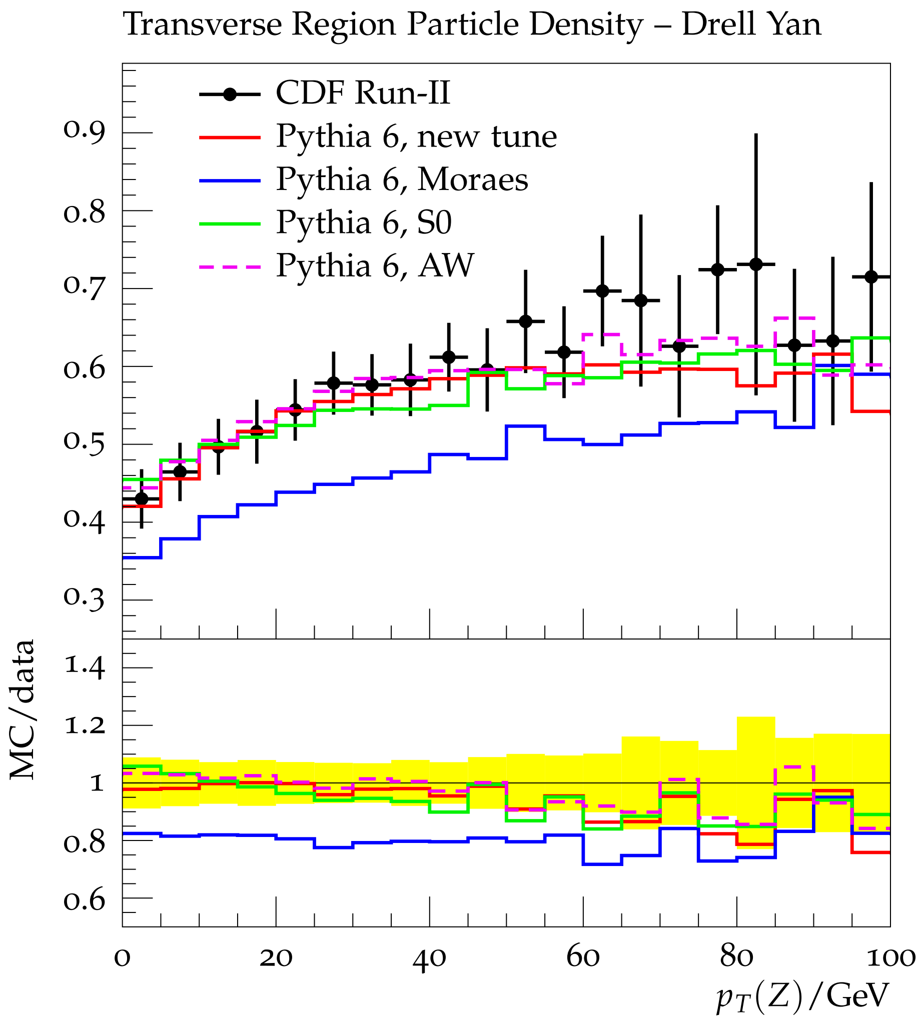
Pythia 6 – Tevatron Comparisons



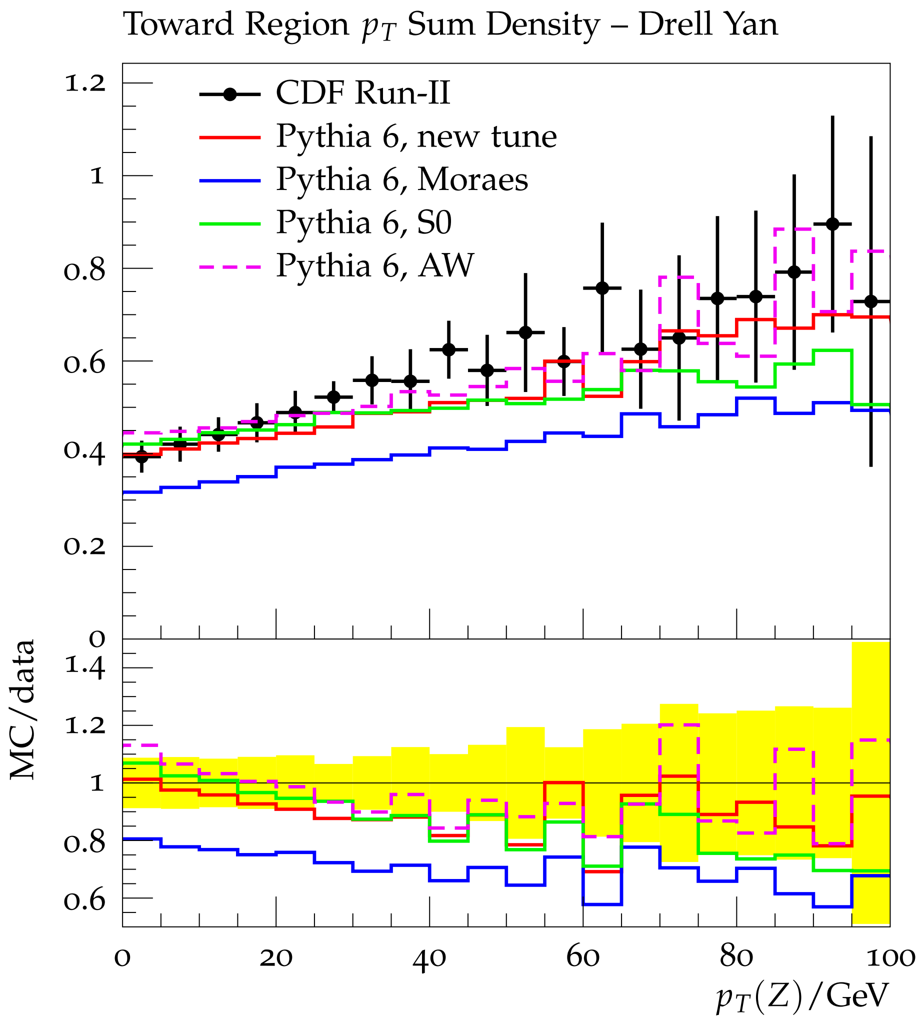


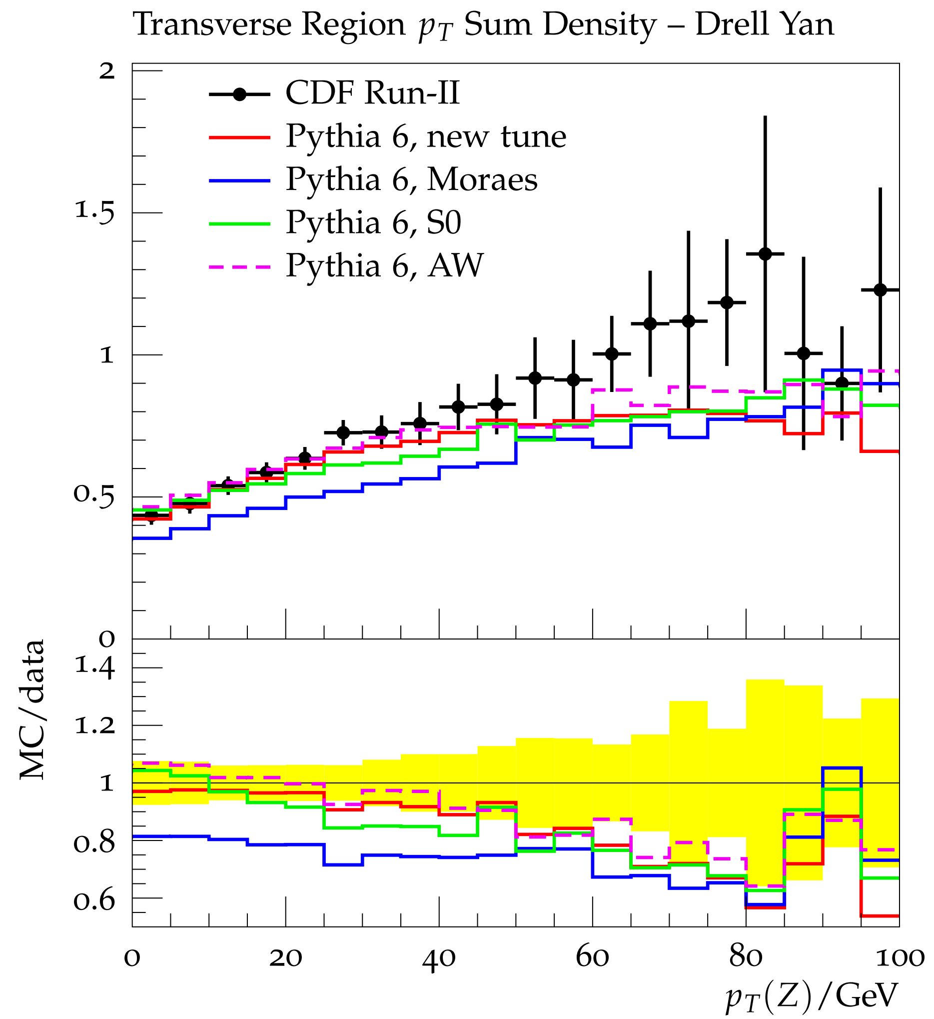
Pythia 6 – Tevatron Comparisons



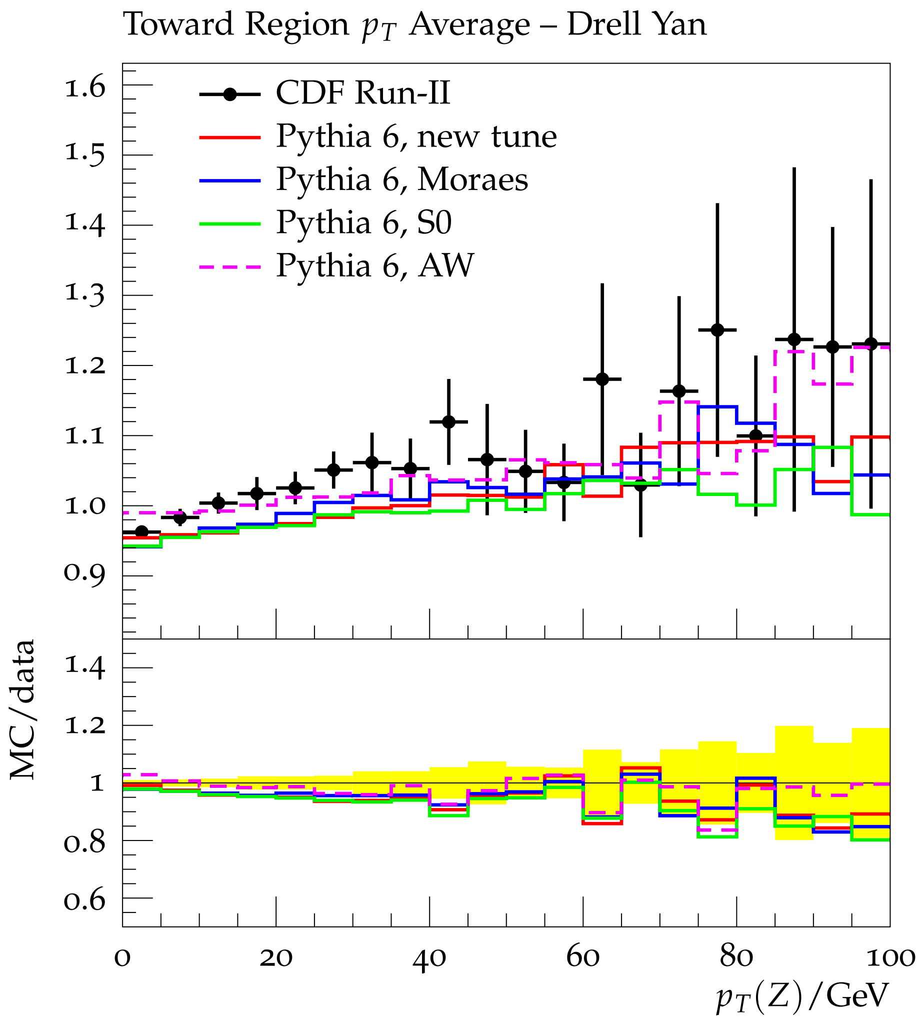


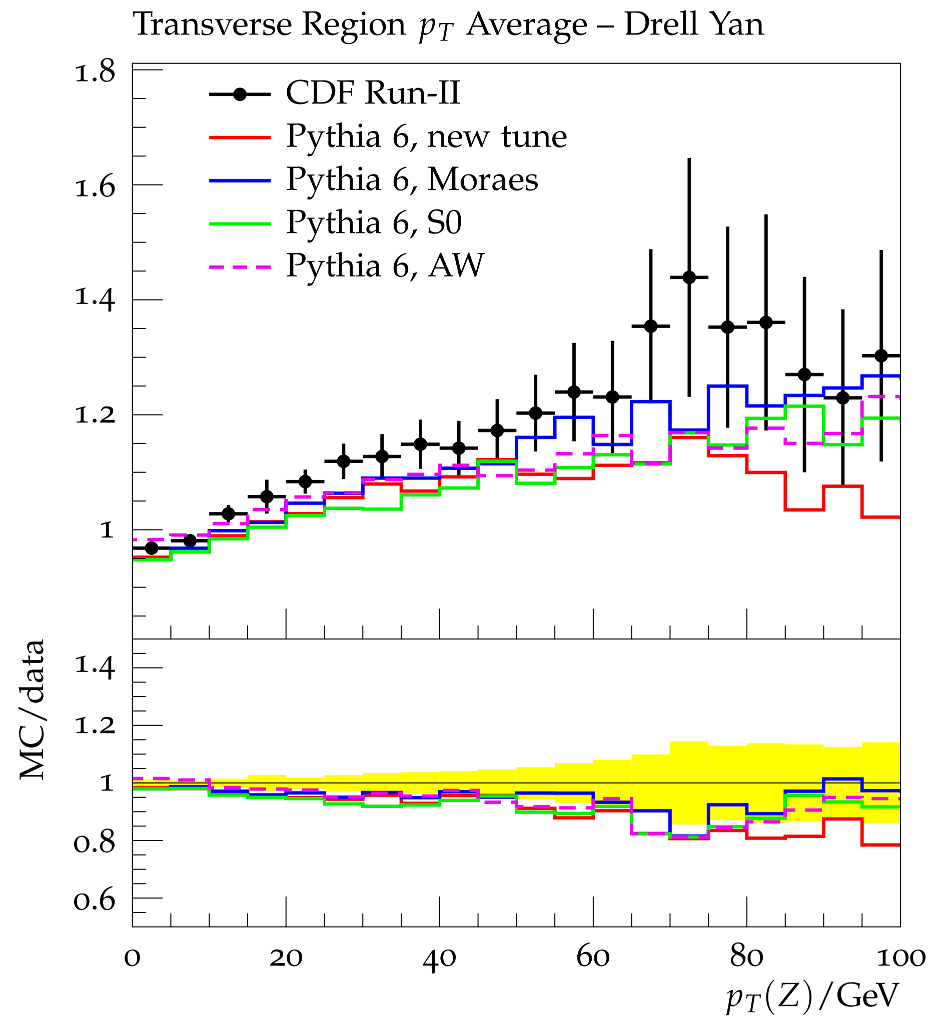
Pythia 6 – Tevatron Comparisons



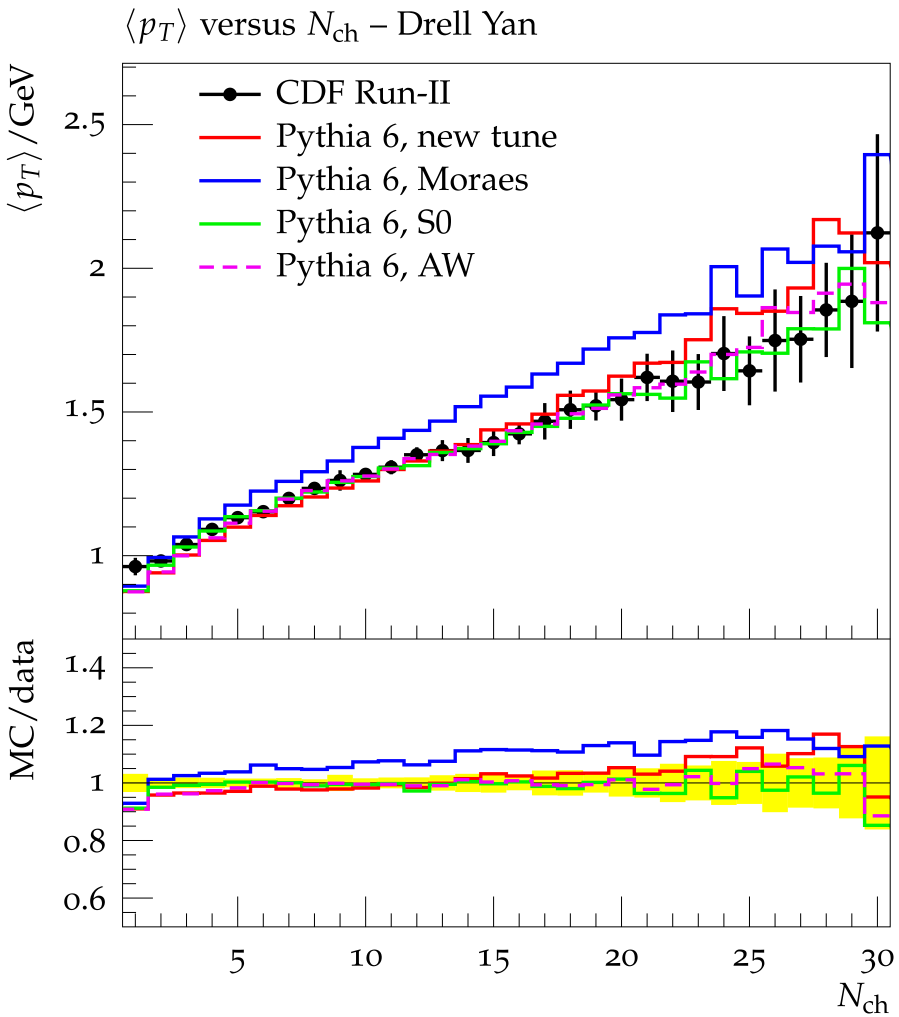


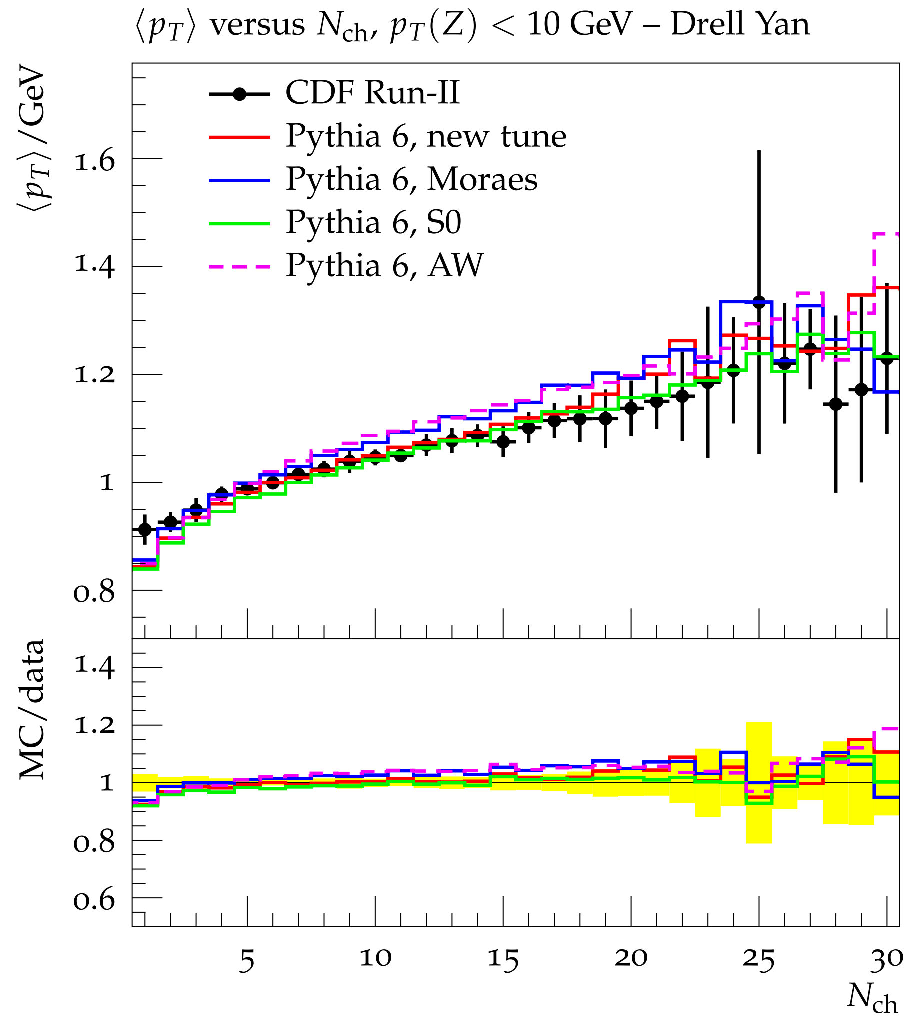
Pythia 6 – Tevatron Comparisons





Pythia 6 – Tevatron Comparisons





Herwig

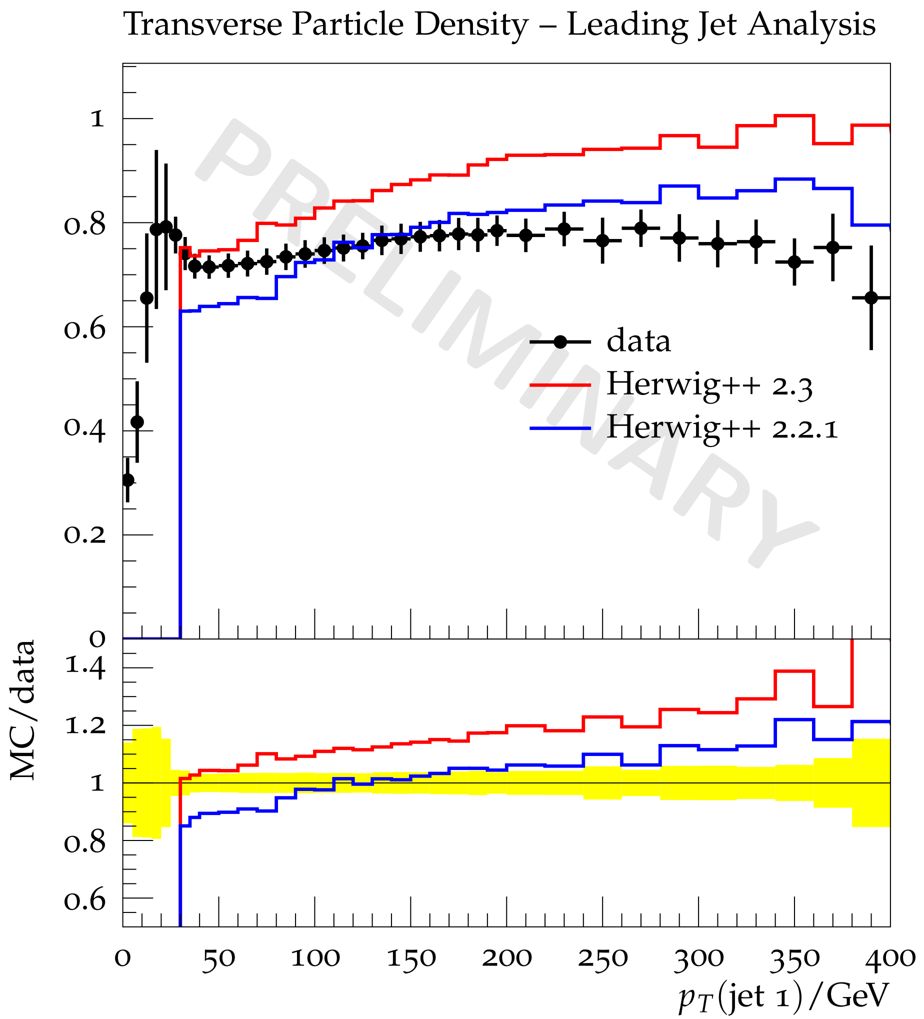
Fortran Herwig:

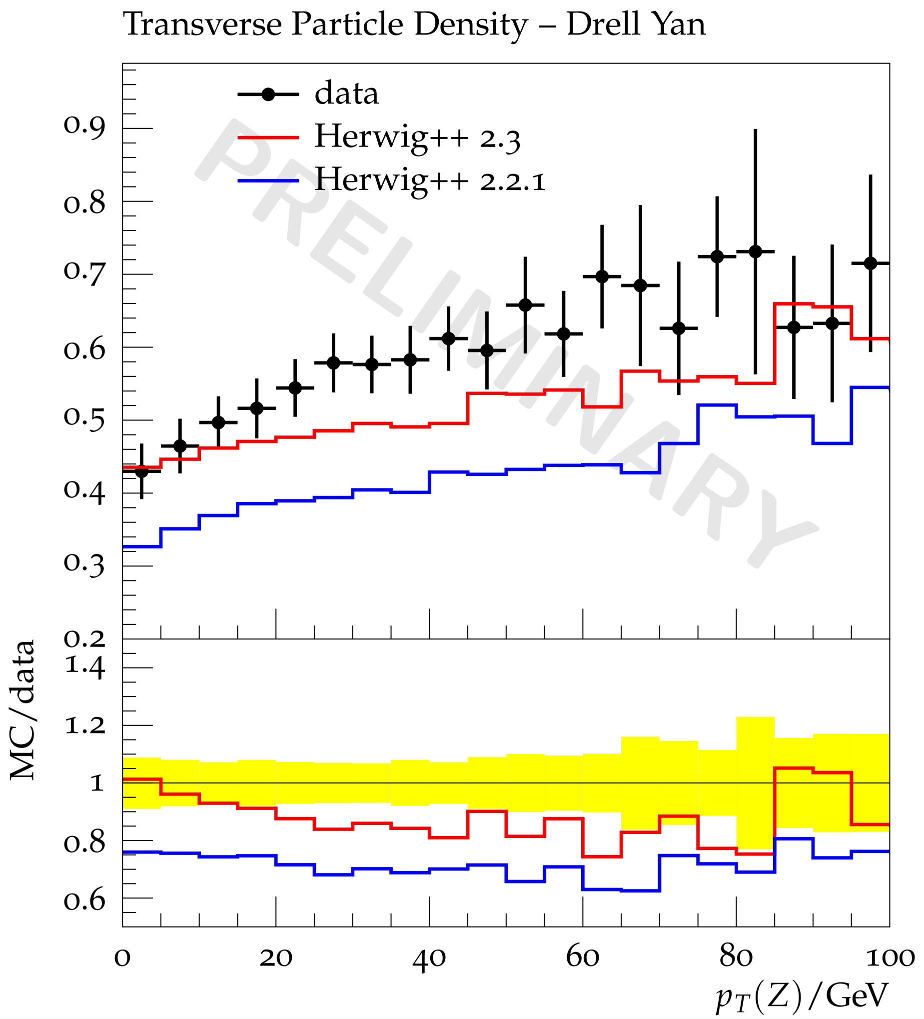
- Shown today is the last version, with Jimmy and default settings.
- Untuned.
- Not able to produce min-bias.
- Unsupported and depreciated $\rightarrow$ use Herwig++.

Herwig++:

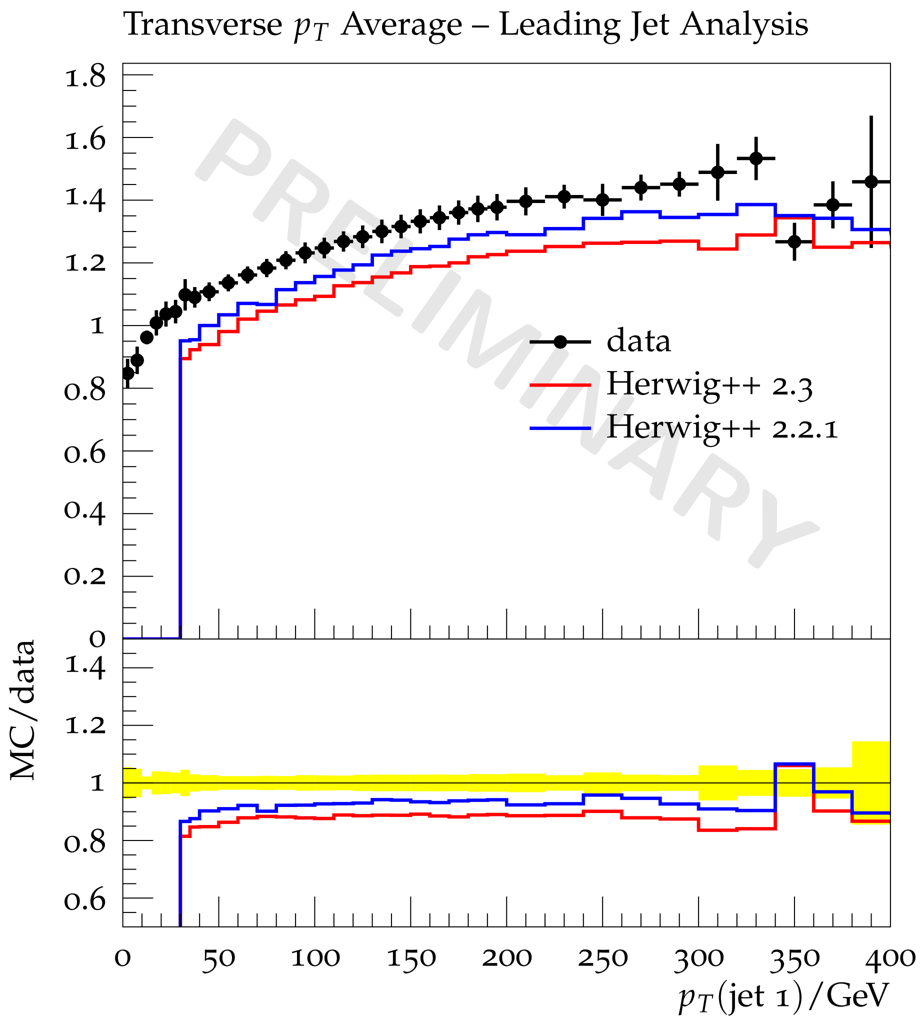
- Stable version 2.2.1, using new multiple scattering MPI model.
- Brute-force tuned.
- Not able to produce min-bias in 2.2.1, but 2.3 will include it.

Herwig++

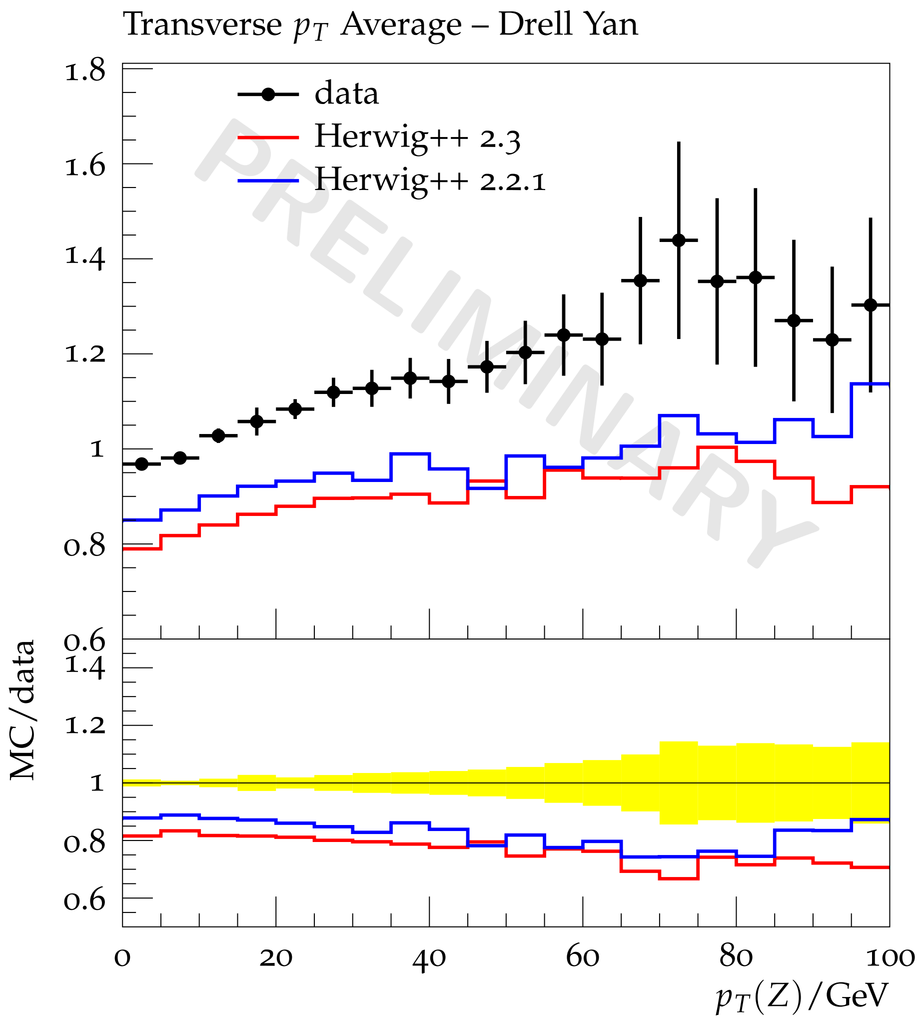

Transverse Particle Density – Leading Jet Analysis
PRELIMINARY
data
Herwig++ 2.3
Herwig++ 2.2.1
MC/data
p_T(jet 1)/GeV


Transverse Particle Density – Drell Yan
PRELIMINARY
data
Herwig++ 2.3
Herwig++ 2.2.1
MC/data
$p_T(Z)$/GeV

Herwig++



Transverse p_T Average – Leading Jet Analysis
PRELIMINARY
data
Herwig++ 2.3
Herwig++ 2.2.1
MC/data
p_T(jet 1)/GeV

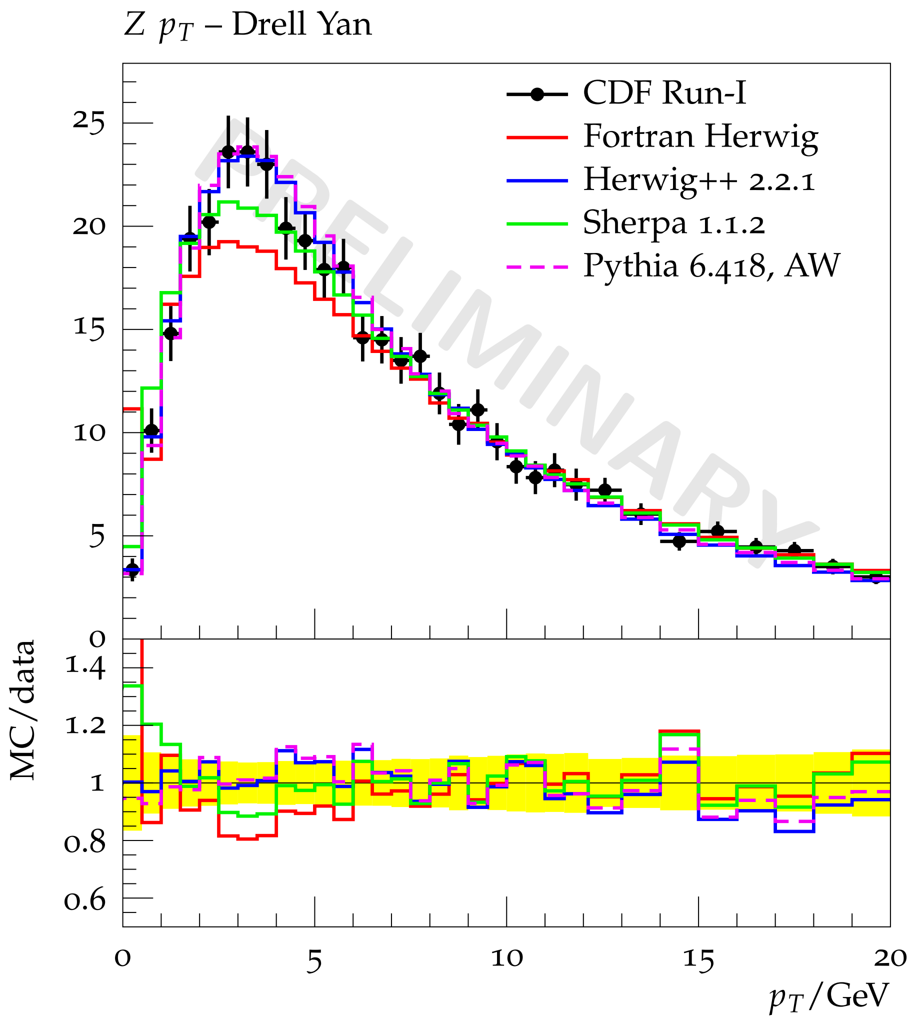


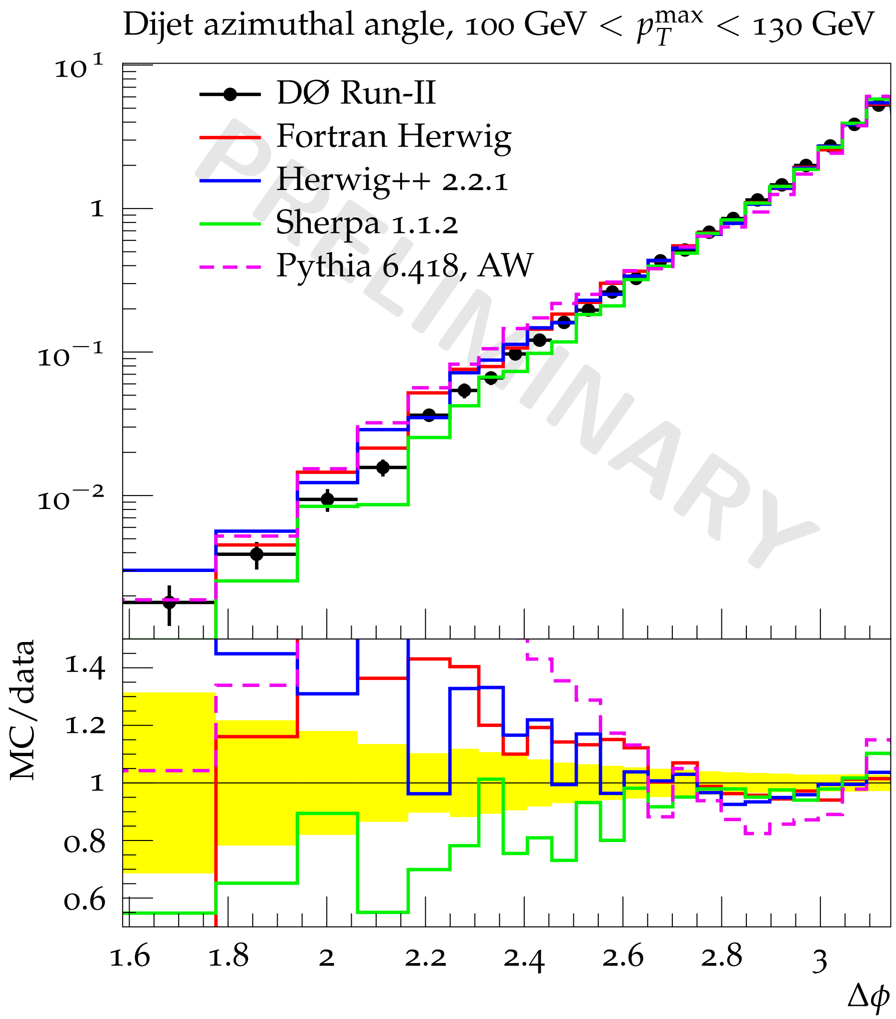
Transverse p_T Average – Drell Yan
data
Herwig++ 2.3
Herwig++ 2.2.1
PRELIMINARY
MC/data
$p_T(Z)$/GeV

Sherpa

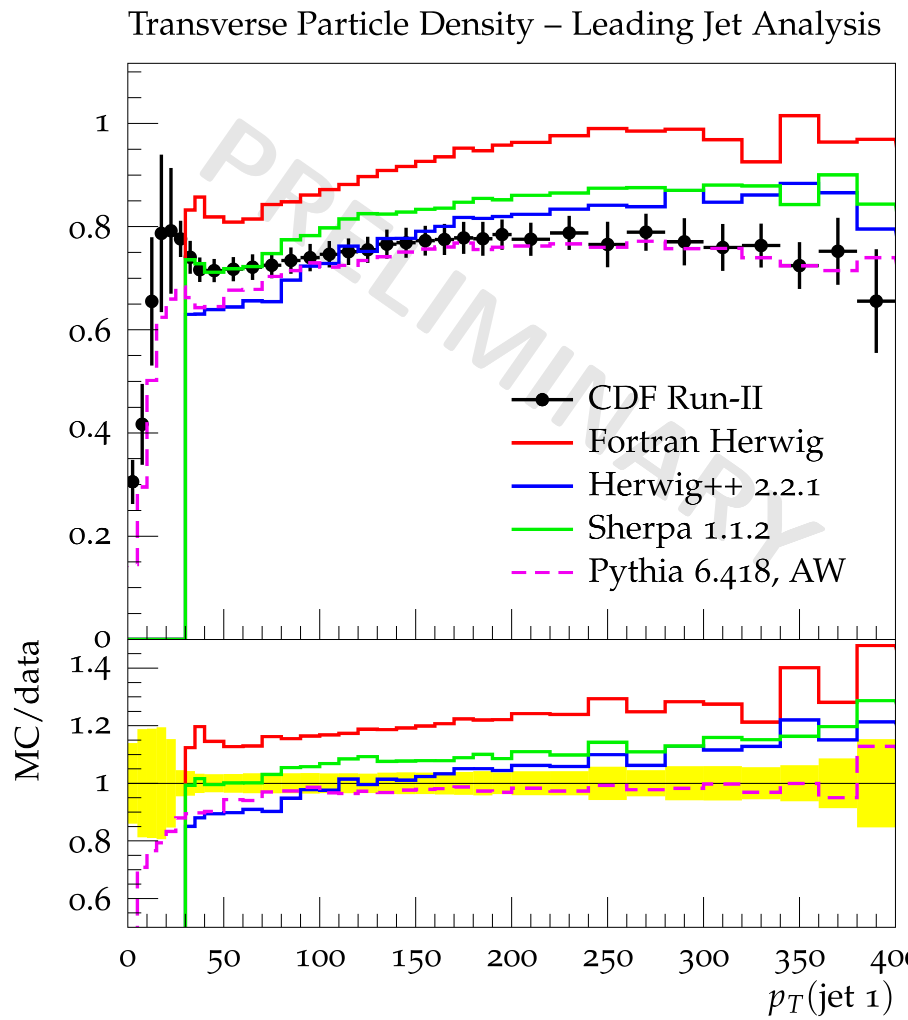
- Shown today is version 1.1.2
- QCD plots: up to three jets from ME
- DY plots: e^+e^- plus up to two jets from ME
- Cluster hadronization – still under heavy development
- Coarse-tuned “by eye”
- Not able to produce min-bias (can go down to $\sim$ 5 GeV)

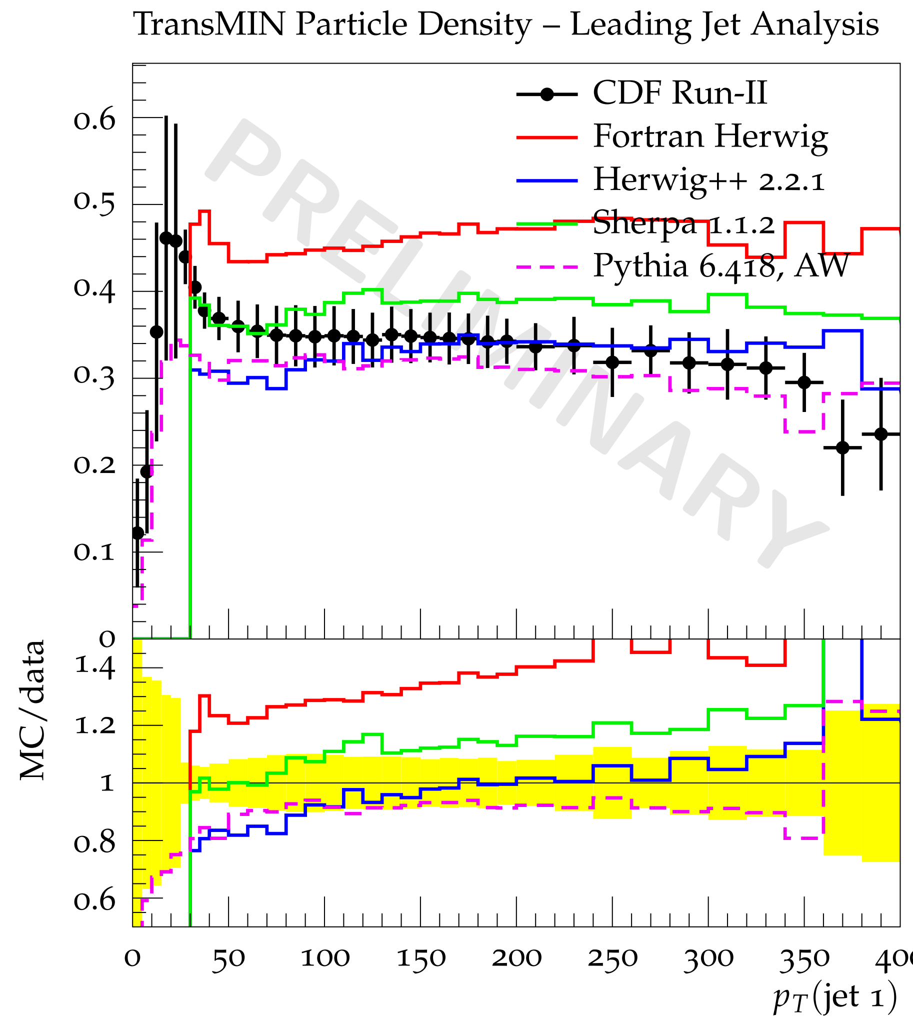
Herwig, Sherpa – Tevatron Comparisons



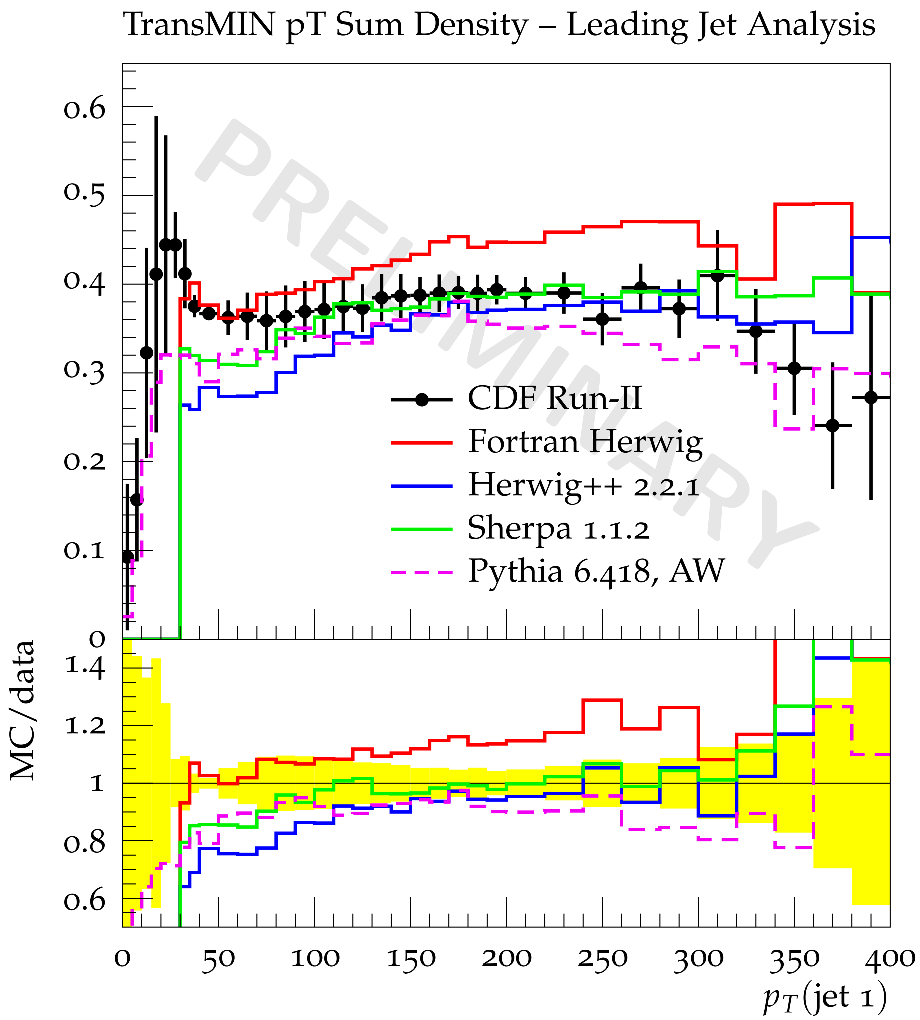


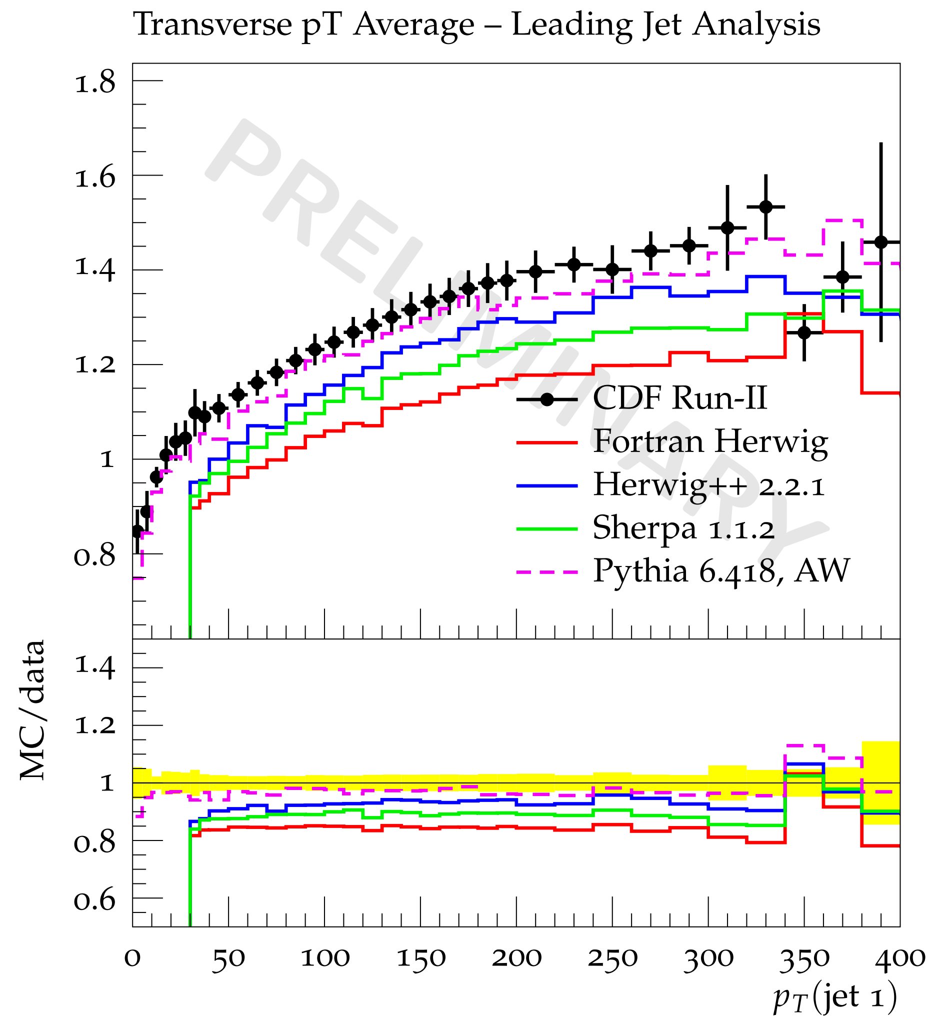
Herwig, Sherpa – Tevatron Comparisons



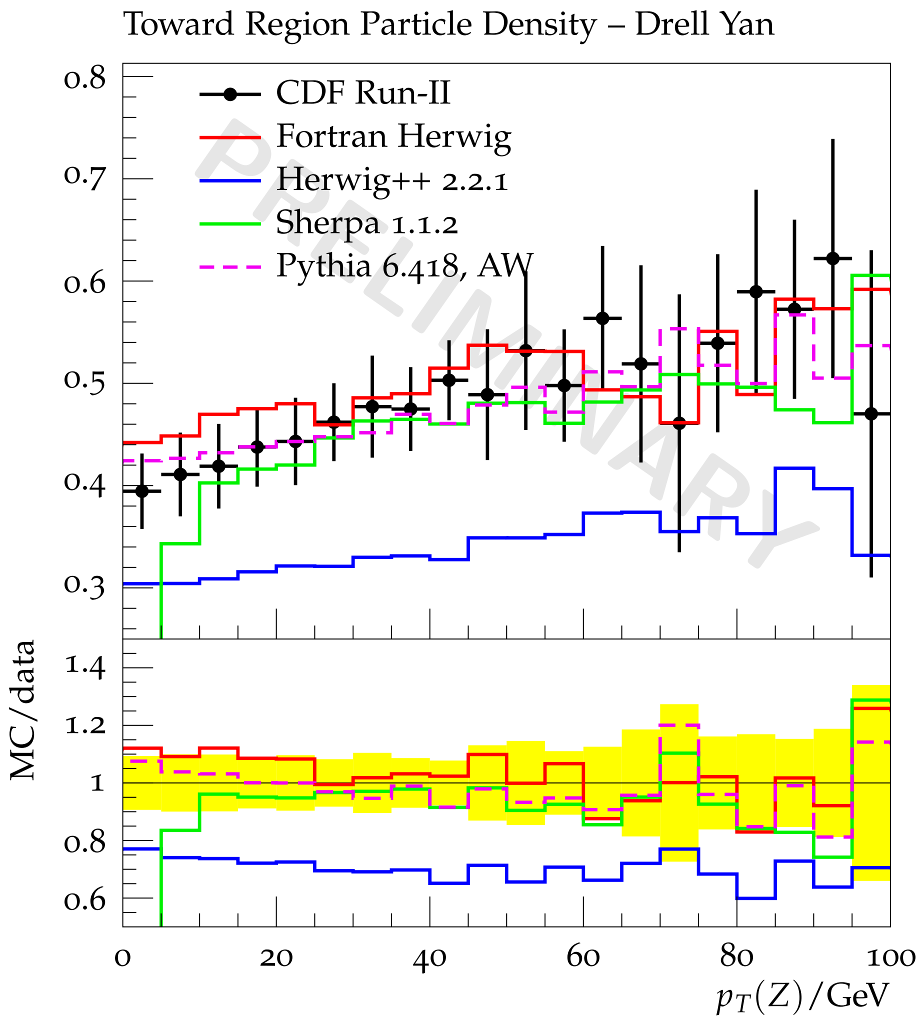


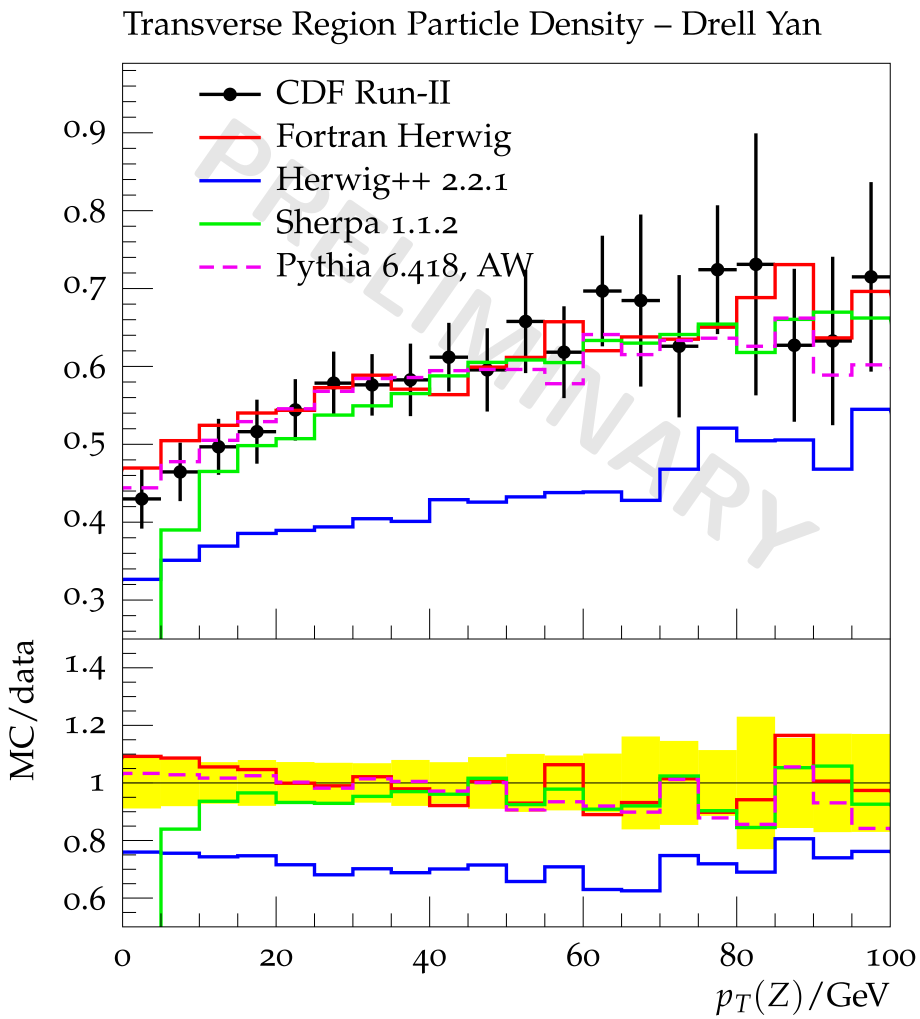
Herwig, Sherpa – Tevatron Comparisons



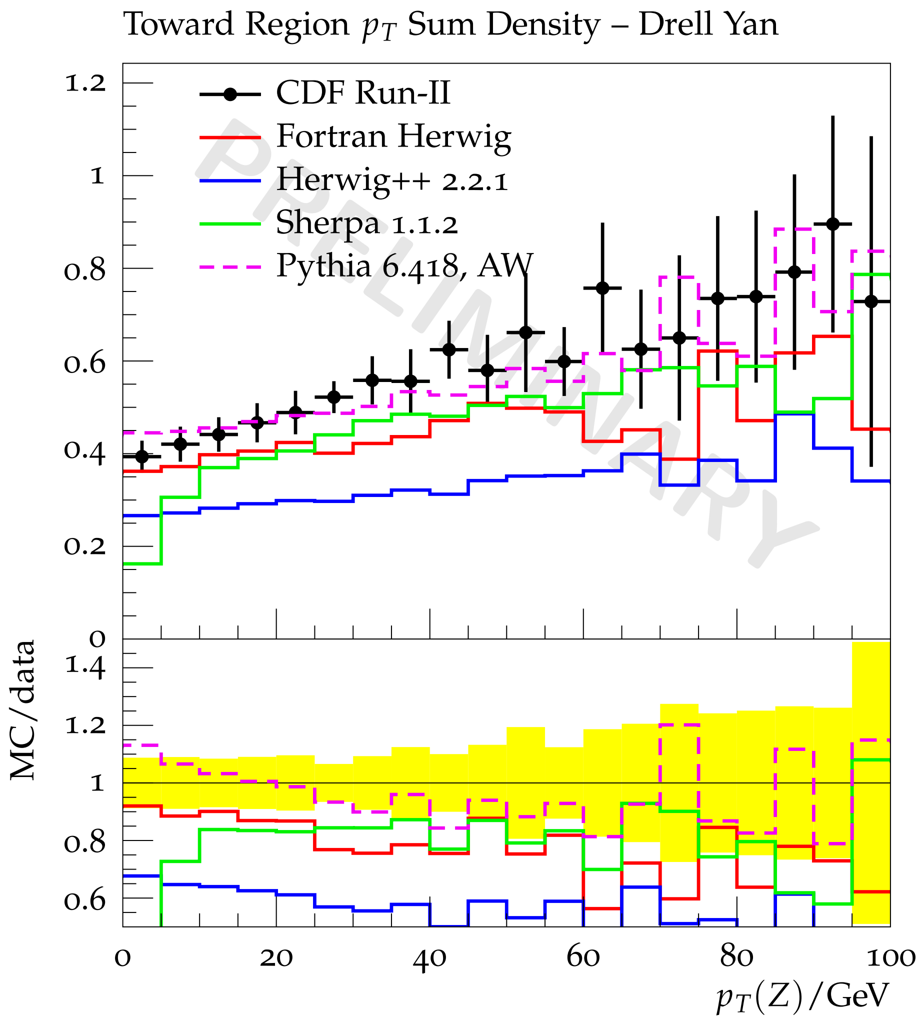


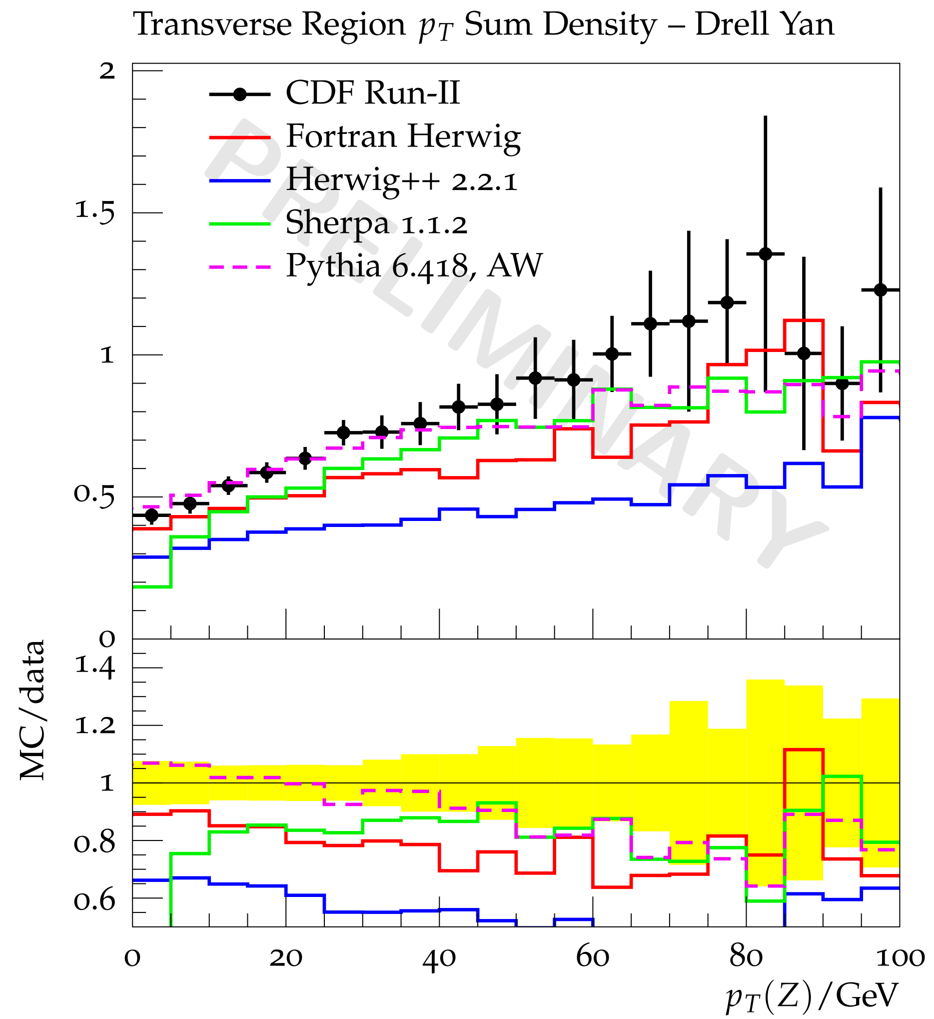
Herwig, Sherpa – Tevatron Comparisons



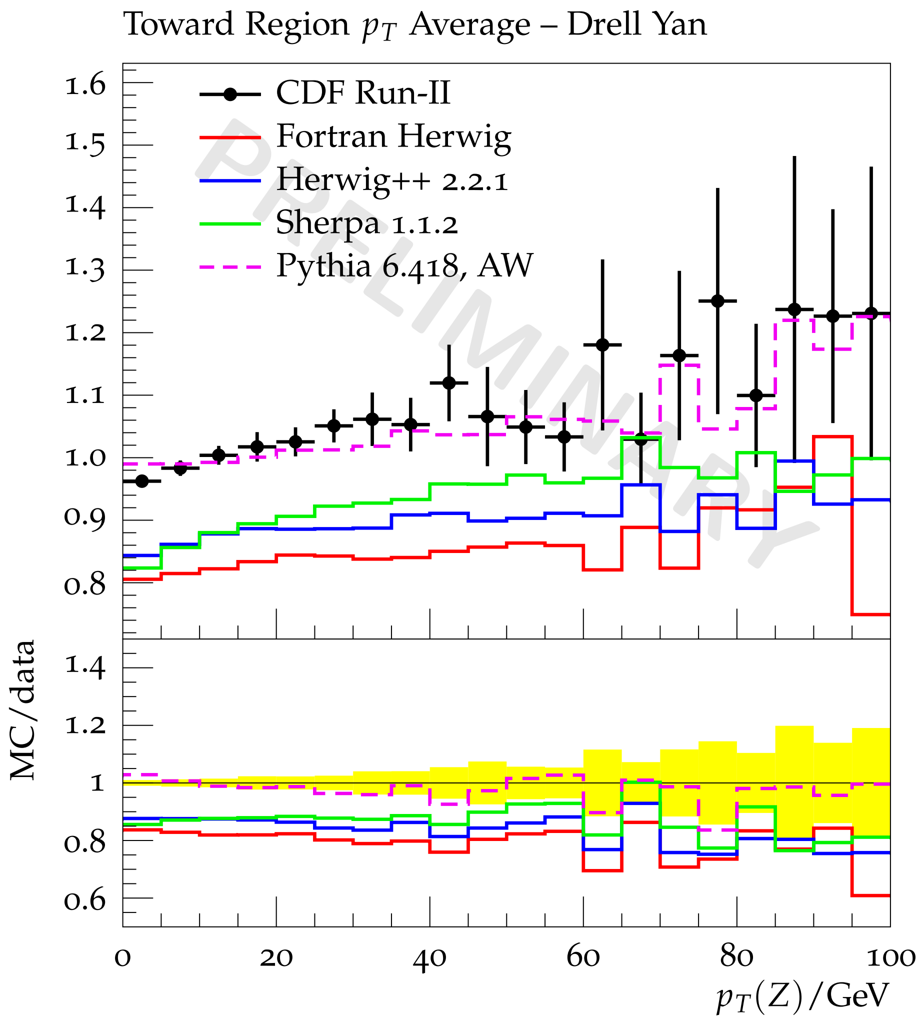


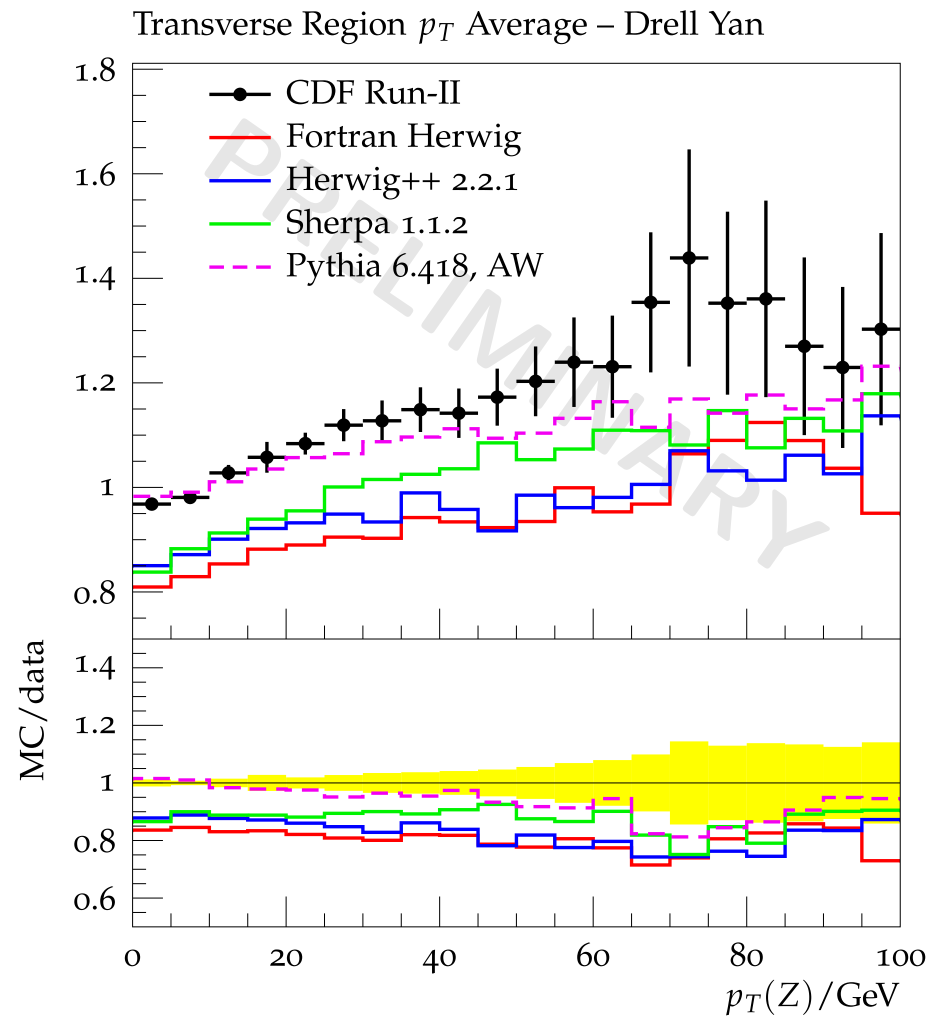
Herwig, Sherpa – Tevatron Comparisons



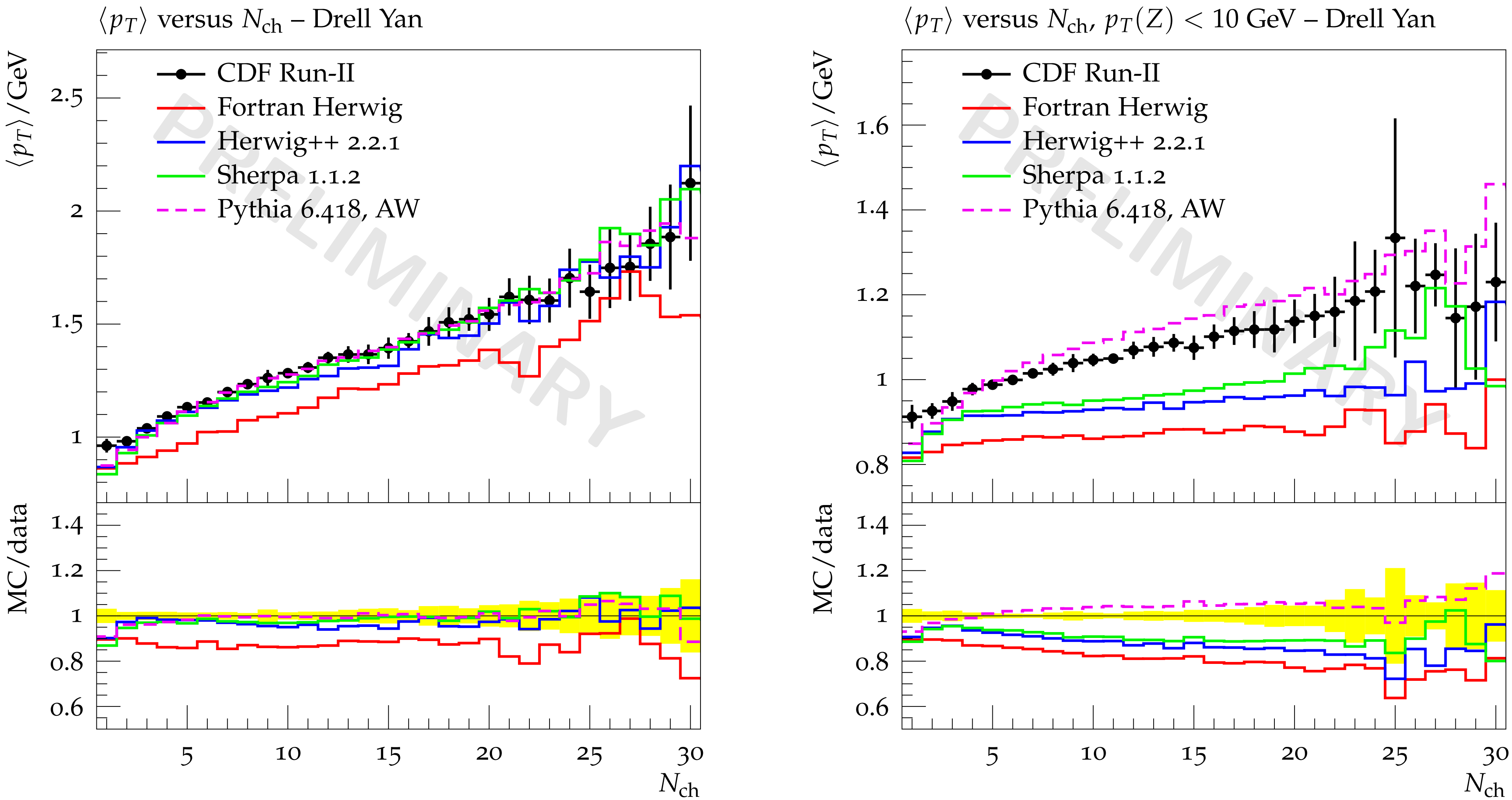


Herwig, Sherpa – Tevatron Comparisons





Herwig, Sherpa – Tevatron Comparisons



Outlook

Things on our list for the (near) future:

- Publish the tunings we already have
- Finish Pythia 6 UE tune with p_T shower and new MPI model
- Tune Pythia 8
- Tune Sherpa: shower, hadronization
- Include more data in the tuning and validation
- Provide side-by-side comparisons of main tunings and generators

Summary

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

Tools for systematic tuning of different event generators have been developed and tested in MCnet. Rivet is a central place to have analyses for MC validation.

Data/MC comparisons show the features and problems of different generators and tunings.

Pythia 6 has been successfully tuned to LEP and Tevatron data (UE tune of new shower and MPI model still in progress).

A Sherpa tuning is in progress.