

Monte Carlo tuning and generator validation

Andy Buckley¹, Hendrik Hoeth^{2†}, Heiko Lacker³, Holger Schulz³, Eike von Seggern³

¹Institute for Particle Physics Phenomenology, Durham University, UK

²Department of Theoretical Physics, Lund University, Sweden

³Physics Department, Berlin Humboldt University, Germany

Abstract

We present the Monte Carlo generator tuning strategy followed, and the tools developed, by the MCnet CEDAR project. We also present new tuning results for the Pythia 6.4 event generator which are based on event shape and hadronisation observables from e^+e^- experiments, and on underlying event and minimum bias data from the Tevatron. Our new tunes are compared to existing tunes and to Peter Skands' new "Perugia" tunes.

1 Introduction

With the LHC starting soon, collider based particle physics is about to enter a new energy regime. Everybody is excited about the possibilities of finding new physics beyond the TeV scale, but the vast majority of events at the LHC will be Standard Model QCD events. The proton will be probed at low Björken x where current PDF fits have large uncertainties, jets above 1 TeV will be seen, and the behaviour of the pp total cross-section and multiple parton interactions will be measured at values of \sqrt{s} where extrapolation from current data is challenging. No discoveries of new physics can be claimed before the Standard Model at these energies is measured and understood.

Monte Carlo event generators play an important role in virtually every physics analysis at collider experiments. They are used to evaluate signal and background events, and to design the analyses. It is essential that the simulations describe the data as accurately as possible. The main point here is not to focus on just one or two distributions, but to look at a wide spectrum of observables. Only if the Monte Carlo agrees with many complementary observables can we trust it to have predictive power, and from disagreements we can learn something about model deficiencies and the underlying physics.

As Monte Carlo event generators are based on phenomenological models and approximations, there are a number of parameters that need to be tweaked if the generator is to describe the experimental data. In the first part of this talk we present a strategy for systematic Monte Carlo parameter tuning. In the second part two new tunes of the Pythia 6.4 generator [1] are presented and compared to other tunings.

2 MC tuning

Every Monte Carlo event generator has a number of relatively free parameters which must be tuned to make the generator describe experimental data in the best possible way. Such param-

[†]speaker

ters can be found almost everywhere in Monte Carlo generators – all the way from the (perturbative) hard interaction to the (non-perturbative) hadronisation process. Naturally the majority of parameters are found in the non-perturbative physics models.

While all the parameters have a physical motivation in their models, there are usually only rough arguments about their scale. Other parameters are measured experimentally (like α_s), but as the Monte Carlo event generators use them in a fixed-order scheme (unlike nature) they need to be adjusted, too.

Going through the steps of event generation and identifying the most important parameters, one typically finds $\mathcal{O}(20\text{--}30)$ parameters of particular importance to collider experiments. Most of these parameters are highly correlated in a non-trivial way. We can group the parameters in approximately independent sets e. g. in flavour, fragmentation, and underlying event parameters, to reduce the number to be optimised against any single set of observables. Nevertheless, the number of parameters to be simultaneously tuned is $\mathcal{O}(10)$. A manual or brute-force approach to Monte Carlo tuning is not very practical: it is very slow, and manual tunings in particular depend very much on the experience of the person performing the tuning (at the same time there is a strong anti-correlation between experience and willingness to produce a new tune manually).

2.1 A systematic tuning strategy

In this talk, we describe the Professor tuning system, which eliminates the problems with manual and brute-force tunings by parameterising a generator’s response to parameter shifts on a bin-by-bin basis, a technique introduced by the Delphi-collaboration [2, 3]. This parameterisation, unlike a brute-force method, is then amenable to numerical minimisation within a timescale short enough to make explorations of tuning criteria possible.

2.1.1 Predicting the Monte Carlo output

The first step of any tuning is to define the parameters that shall be varied, together with the variation intervals. This requires a thorough understanding of the generator’s model, its parameters and the available data – all the relevant parameters for a certain model should enter the tuning, but none of the irrelevant ones. A fragmentation tune for example must include the shower cut-off parameter, while a tune of the flavour composition had better not be dependent on it.

Once we have settled on a set of parameter intervals, it is time to obtain a predictive function for the Monte Carlo output. Actually we generate an ensemble of such functions. For each observable bin b a polynomial is fitted to the Monte Carlo response MC_b to changes in the parameter vector $\vec{p} = (p_1, \dots, p_P)$ of the P parameters varied in the tune. To account for lowest-order parameter correlations, a polynomial of at least second-order is used as the basis for bin parameterisation:

$$\text{MC}_b(\vec{p}) \approx f^{(b)}(\vec{p}) = \alpha_0^{(b)} + \sum_i \beta_i^{(b)} p_i + \sum_{i \leq j} \gamma_{ij}^{(b)} p_i p_j \quad (1)$$

We have tested this to give a good approximation of the true Monte Carlo response for real-life observables.

The number of parameters and the order of the polynomial fix the number of coefficients to be determined. For a second order polynomial in P parameters, the number of coefficients is

$$N_2^{(P)} = 1 + P + P(P + 1)/2, \quad (2)$$

since only the independent components of the matrix term are to be counted.

Given a general polynomial, we must now determine the coefficients α, β, γ for each bin so as to best mimic the true generator behaviour. This could be done by a Monte Carlo numerical minimisation method, but there would be a danger of finding sub-optimal local minima, and automatically determining convergence is a potential source of problems. Fortunately, this problem can be cast in such a way that a deterministic method can be applied.

One way to determine the polynomial coefficients would be to run the generator at as many parameter points, N , as there are coefficients to be determined. A square $N \times N$ matrix can then be constructed, mapping the appropriate combinations of parameters on to the coefficients to be determined; a normal matrix inversion can then be used to solve the system of simultaneous equations and thus determine the coefficients. Since there is no reason for the matrix to be singular, this method will always give an “exact” fit of the polynomial to the generator behaviour. However, this does not reflect the true complexity of the generator response: we have engineered the exact fit by restricting the number of samples on which our interpolation is based, and it is safe to assume that taking a larger number of samples would show deviations from what a polynomial can describe, both because of intrinsic complexity in the true response function and because of the statistical sampling error that comes from running the generator for a finite number of events. What we would like is to find a set of coefficients (for each bin) which average out these effects and are a least-squares best fit to the oversampled generator points. As it happens, there is a generalisation of matrix inversion to non-square matrices – the *pseudoinverse* [4] – with exactly this property.

As suggested, the set of anchor points for each bin are determined by randomly sampling the generator from N parameter space points in the P -dimensional parameter hypercube $[\vec{p}_{\min}, \vec{p}_{\max}]$ defined by the user. This definition requires physics input – each parameter p_i should have its upper and lower sampling limits $p_{\min, \max}$ chosen so as to encompass all reasonable values; we find that generosity in this definition is sensible, as Professor may suggest tunes which lie outside conservatively chosen ranges, forcing a repeat of the procedure. On the other hand the parameter range should not be too large, in order to keep the volume of the parameter space small and to make sure that the parabolic approximation gives a good fit to the true Monte Carlo response. Each sampled point may actually consist of many generator runs, which are then merged into a single collection of simulation histograms. The simultaneous equations solution described above is possible if the number of sampled points is the same as the number of coefficients between the P parameters, i.e. $N = N_{\min}^{(P)} = N_n^{(P)}$. The more robust pseudoinverse method applies when $N > N_{\min}^{(P)}$: we prefer to oversample by at least a factor of 2. The numerical implementation of the pseudoinverse uses a standard singular value decomposition (SVD) [5].

2.1.2 Comparing to data and optimising the parameters

With the functions $f^{(b)}(\vec{p})$ we now have a very fast way of predicting the behaviour of the Monte Carlo generator. To get the Monte Carlo response for any parameter setting inside the defined parameter hypercube it is not necessary anymore to run the generator, but we can simply evaluate the polynomial. This allows us to define a goodness of fit function comparing data and (approximated) Monte Carlo which can be minimised in a very short time.

We choose a heuristic χ^2 function, but other goodness of fit (GoF) measures can certainly be used. Since the relative importance of various distributions in the observable set is a subjective thing – given 20 event shape distributions and one charged multiplicity, it is certainly sensible to weight up the multiplicity by a factor of at least 10 or so to maintain its relevance to the GoF measure – we include per-observable weights, $w_{\mathcal{O}}$ for each observable \mathcal{O} , in our χ^2 definition:

$$\chi^2(\vec{p}) = \sum_{\mathcal{O}} w_{\mathcal{O}} \sum_{b \in \mathcal{O}} \frac{(f^{(b)}(\vec{p}) - \mathcal{R}_b)^2}{\Delta_b^2}, \quad (3)$$

where \mathcal{R}_b is the reference (i. e. data) value for bin b and the total error Δ_b is the sum in quadrature of the reference error and the statistical generator errors for bin b . In practice we attempt to generate sufficient events at each sampled parameter point that the statistical MC error is much smaller than the reference error for all bins.

It should be noted that there is unavoidable subjectivity in the choice of these weights, and a choice of equal weights is no more sensible than a choice of uniform priors in a Bayesian analysis; physicist input is necessary in both choosing the admixture of observable weights according to the criteria of the generator audience – a b -physics experiment may prioritise distributions that a general-purpose detector collaboration would have little interest in – and to ensure that the end result is not overly sensitive to the choice of weights.

The final stage is to minimise the parameterised χ^2 function. It is tempting to think that there is scope for an analytic global minimisation at this order of polynomial, but not enough Hessian matrix elements may be calculated to constrain all the parameters and hence we must finally resort to a numerical minimisation. This is the numerically weakest point in the method, as the weighted quadratic sum of hundreds of polynomials is a very complex function and there is scope for getting stuck in a non-global minimum. Hence the choice of minimiser is important.

The output from the minimisation is a vector of parameter values which, if the parameterisation and minimisation stages are faithful, should be the optimal tune according to the (subjective) criterion defined by the choice of observable weights.

2.2 Tools

We have implemented the tuning strategy described above in the Professor software package. Professor reads in Monte Carlo and data histogram files, parameterises the Monte Carlo response, and performs the χ^2 minimisation.

The Monte Carlo histograms used as input for Professor are generated with Rivet [6]. Rivet is an analysis framework for Monte Carlo event generator validation. By reading in HepMC event records, Rivet can be used with virtually all common event generators, and this well-defined interface between generator and analysis tool ensures that the physics analyses are implemented

Parameter	Pythia 6.418 default	Final tune	
PARJ(1)	0.1	0.073	diquark suppression
PARJ(2)	0.3	0.2	strange suppression
PARJ(3)	0.4	0.94	strange diquark suppression
PARJ(4)	0.05	0.032	spin-1 diquark 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	0.63	η suppression
PARJ(26)	0.4	0.12	η' suppression

Table 1: Tuned flavour parameters and their defaults.

in a generator-independent way. A key feature of Rivet is that the reference data can be taken directly from the HepData archive [7] and is used to define the binnings of the Monte Carlo histograms, automatically ensuring that there is no problem with synchronising bin edge positions. At present, there are about 40 key analyses mainly from LEP and Tevatron, but also from SLD, RHIC, PETRA, and other accelerators. More analyses are constantly being added.

3 Tuning Pythia 6.4

For the first production tuning we chose the Pythia 6.4 event generator, as this is a well-known generator which has been tuned before and which we expected to behave well. Naturally the first step in tuning a generator is to fix the flavour composition and the fragmentation parameters to the precision data from LEP and SLD before continuing with the parameters related to hadron collisions, for which we use data from the Tevatron.

3.1 Parameter factorisation strategy

In Pythia the parameters for flavour composition decouple well from the non-flavour hadronisation parameters such as a , b , σ_q , or the shower parameters (α_s , cut-off). Parameters related to the underlying event and multiple parton interactions are decoupled from the flavour and fragmentation parameters. In order to keep the number of simultaneously tuned parameters small, we decided to follow a three-stage strategy. In the first step the flavour parameters were optimised, keeping almost everything else at its default values (including using the virtuality-ordered shower). In the second step the non-flavour hadronisation and shower parameters were tuned – using the optimised flavour parameters obtained in the first step. The final step was tuning the underlying event and multiple parton interaction parameters to data from CDF and DØ.

3.2 Flavour parameter optimisation

The observables used in the flavour tune were hadron multiplicities and their ratios with respect to the π^+ multiplicity measured at LEP 1 and SLD [8], as well as the b -quark fragmentation function measured by the Delphi collaboration [9], and flavour-specific mean charged multiplicities as

measured by the Opal collaboration [10]. For this first production we chose to use a separate tuning of the Lund-Bowler fragmentation function for b -quarks (invoked in Pythia 6.4 by setting $\text{MSTJ}(11) = 5$) with a fixed value of $r_b = 0.8$ ($\text{PARJ}(47)$), as first tests during the validation phase of the Professor framework showed that this setting yields a better agreement with data than the default common Lund-Bowler parameters for c and b quarks.

For the tuning we generated 500k events at each of 180 parameter points. The tuned parameters are the basic flavour parameters like diquark suppression, strange suppression, or spin-1 meson rates. All parameters are listed in Tab. 1 together with the tuning results.

Since the virtuality-ordered shower was used for tuning the flavour parameters, we tested our results also with the p_\perp -ordered shower in order to check if a separate tuning was necessary. Turning on the p_\perp -ordered shower and setting $\Lambda_{\text{QCD}} = 0.23$ (the recommended setting before our tuning effort) we obtained virtually the same multiplicity ratios as with the virtuality-ordered shower. This confirms the decoupling of the flavour and the fragmentation parameters and no re-tuning of the flavour parameters with the p_\perp -ordered shower is needed.

3.3 Fragmentation optimisation

Based on the new flavour parameter settings the non-flavour hadronisation and shower parameters were tuned, separately for the virtuality-ordered and for the p_\perp -ordered shower. The observables used in this step of the tuning were event shape variables, momentum spectra, and the mean charged multiplicity measured by the Delphi collaboration [3], momentum spectra and flavour-specific mean charged multiplicities measured by the Opal collaboration [10], and the b -quark fragmentation function measured by the Delphi collaboration [9].

We tuned the same set of parameters for both shower types (Tab. 2). To turn on the p_\perp -ordered shower, $\text{MSTJ}(41)$ was set to 12 – in the case of the virtuality-ordered shower, this parameter stayed at its default value. For both tunes, we generated 1M events at each of 100 parameter points.

During the tuning of the p_\perp -ordered shower it transpired that the fit prefers uncomfortably low values of the shower cut-off $\text{PARJ}(82)$. Since this value needs to be at least $2 \cdot \Lambda_{\text{QCD}}$, and preferably higher, it was manually fixed to 0.8 to keep the parameters in a physically meaningful regime. Then the fit was repeated with the remaining five parameters.

The second issue we encountered with the p_\perp -ordered shower was that the polynomial parameterisation $f^{(b)}$ for the mean charged multiplicity differed from the real Monte Carlo response by about 0.2 particles. This discrepancy was accounted for during the χ^2 minimisation, so that the final result does not suffer from a bias in this observable.

In Fig. 1 some comparison plots between the Pythia default and our new tune of the virtuality-ordered shower are depicted. Even though this shower has been around for many years and Pythia has been tuned before, there still is room for improvement in the default settings.

Fig. 2 shows comparisons of the p_\perp -ordered shower. This shower is a new option in Pythia and has not been tuned systematically before. Nevertheless, the Pythia manual recommends to set Λ_{QCD} to 0.23. This recommendation is ignored by the ATLAS collaboration, so our plots show our new tune, the default with $\Lambda_{\text{QCD}} = 0.23$, and the settings currently used by ATLAS [11].

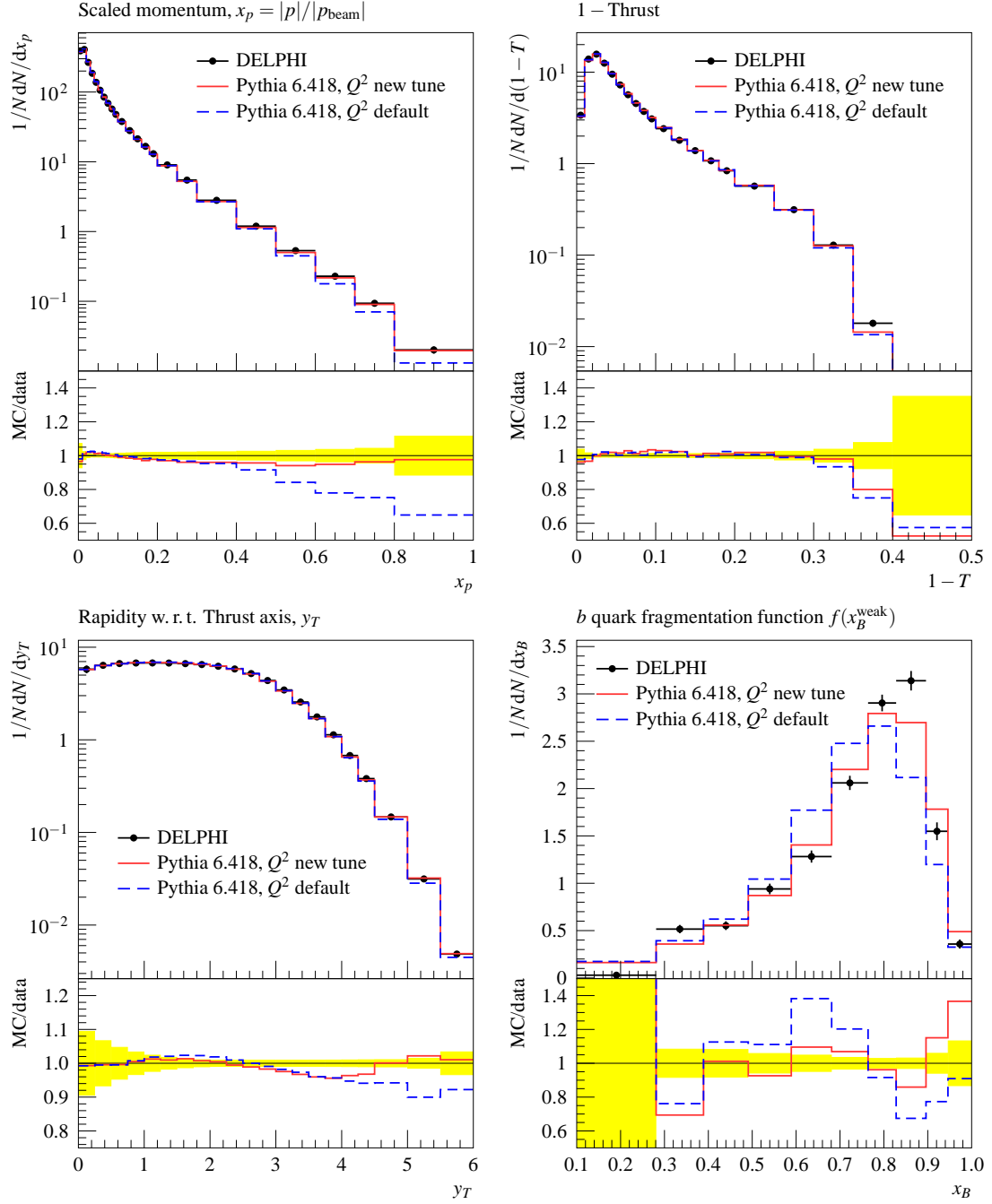


Fig. 1: Some example distributions for e^+e^- collisions using the virtuality-ordered shower. The solid line shows the new tune, the dashed line is the default. Even though the virtuality-ordered shower is well-tested and Pythia has been tuned several times, especially by the LEP collaborations, there is still room for improvement in the default settings. Note the different scale in the ratio plot of the rapidity distribution. The data in these plots has been published by Delphi [3, 9].

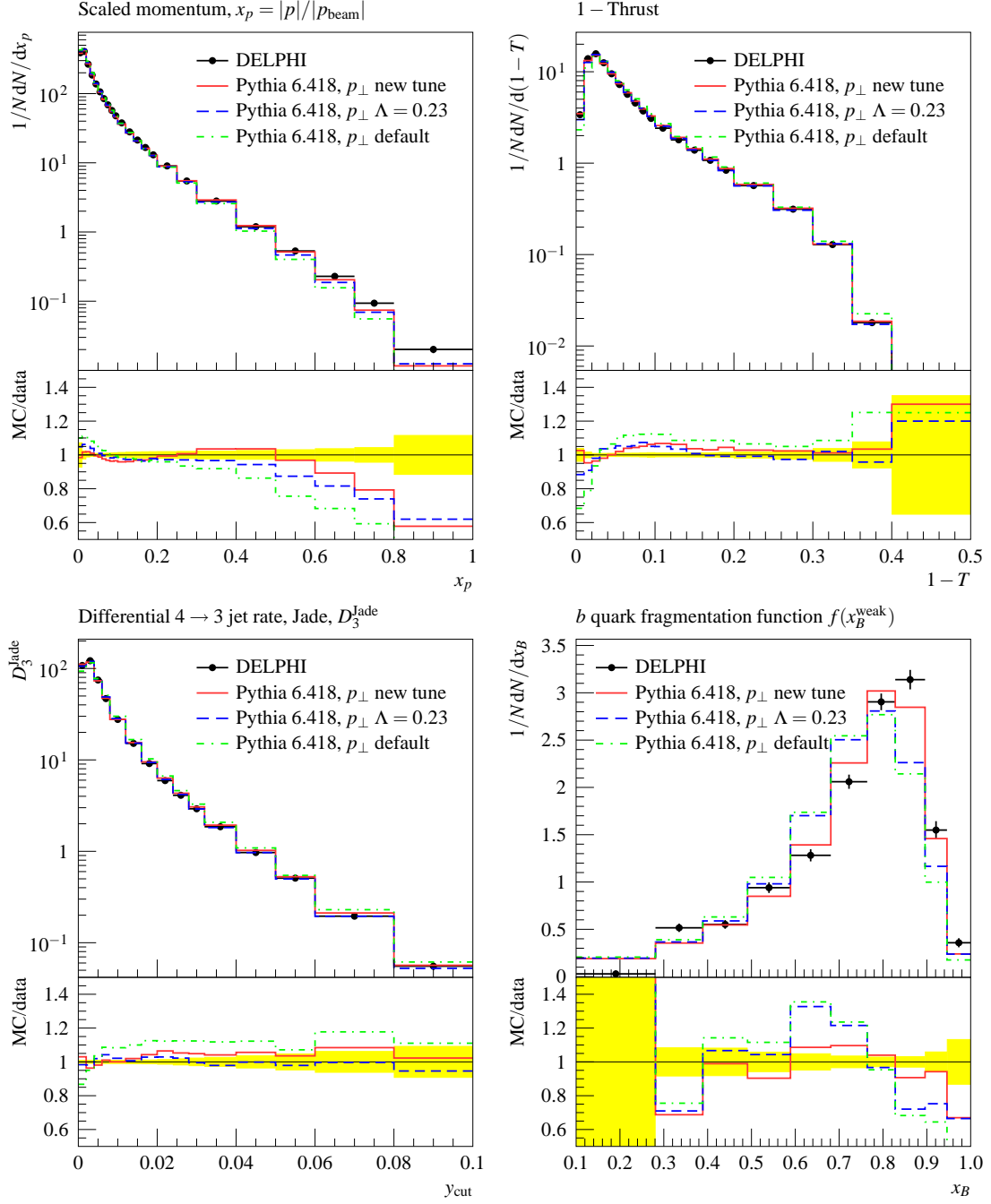


Fig. 2: Some example distributions for e^+e^- collisions using the p_\perp -ordered shower. The solid line shows the new tune, the dashed line is the old recommendation for using the p_\perp -ordered shower (i.e. changing Λ_{QCD} to 0.23), and the dashed-dotted line is produced by switching on the p_\perp -ordered shower leaving everything else at its default. The latter is the unfortunate choice made for the ATLAS-tune. The data has been published by Delphi [3,9].

Parameter	Pythia 6.418 default	Final tune (Q^2)	Final tune (p_\perp)	
MSTJ(11)	4	5	5	frag. function
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	0.67	1.0	r_b
PARJ(81)	0.29	0.29	0.257	Λ_{QCD}
PARJ(82)	1	1.65	0.8	shower cut-off

Table 2: Tuned fragmentation parameters and their defaults for the virtuality and p_\perp -ordered showers.

3.4 Underlying event and multiple parton interactions

For the third step we tuned the parameters relevant to the underlying event, again both for the virtuality-ordered shower and the old MPI model, and for the p_\perp -ordered shower with the interleaved MPI model. This was based on various Drell-Yan, jet physics, and minimum bias measurements performed by CDF and DØ in Run-I and Run-II [12–18].

The new MPI model differs significantly from the old one, hence we had to tune different sets of parameters for these two cases. For the virtuality-ordered shower and old MPI model we took Rick Field’s tune DW [19] as guideline. In the case of the new model we consulted Peter Skands and used a setup similar to his tune S0 [20,21] as starting point. All switches and parameters for the UE/MPI tune, and our results are listed in Tables 3 and 4.

One of the main differences we observed between the models is their behaviour in Drell-Yan physics. The old model had a hard time describing the Z - p_\perp spectrum [12] and we had to assign a high weight to that observable in order to force the Monte Carlo to get the peak region of the distribution right (note that this is the only observable to which we assigned different weights for the tunes of the old and the new MPI model). The new model on the other hand gets the Z - p_\perp right almost out of the box, but underestimates the underlying event activity in Drell-Yan events as measured in [16]. The same behaviour can be observed in Peter Skands’ tunes [22]. We are currently investigating this issue.

Another (albeit smaller) difference shows in the hump of the turn-on in many of the UE distributions in jet physics. This hump is described by the new model, but mostly missing in the old model. Although the origin of this hump is thought to be understood, the model differences responsible for its presence/absence in the two Pythia models is not yet known in any detail.

Figures 3 to 7 show some comparisons between our new tune and various other tunes. For the virtuality-ordered shower with the old MPI model we show Rick Field’s tunes A [23] and DW [19] as references, since they are well-known and widely used. For the p_\perp -ordered shower and the new MPI framework we compare to Peter Skands’ new Perugia0 tune [22]. We also include the current ATLAS tune [11] (even though we don’t believe it has good predictive power¹), since it is widely used at the LHC.

¹Not only is the choice of fragmentation parameters unfortunate (as discussed in Section 3.3) and the tune fails to describe the underlying event in Drell-Yan events, but also the energy scaling behaviour in this tune is pretty much ruled out by the data [24], making it in our eyes a particularly bad choice for LHC predictions.

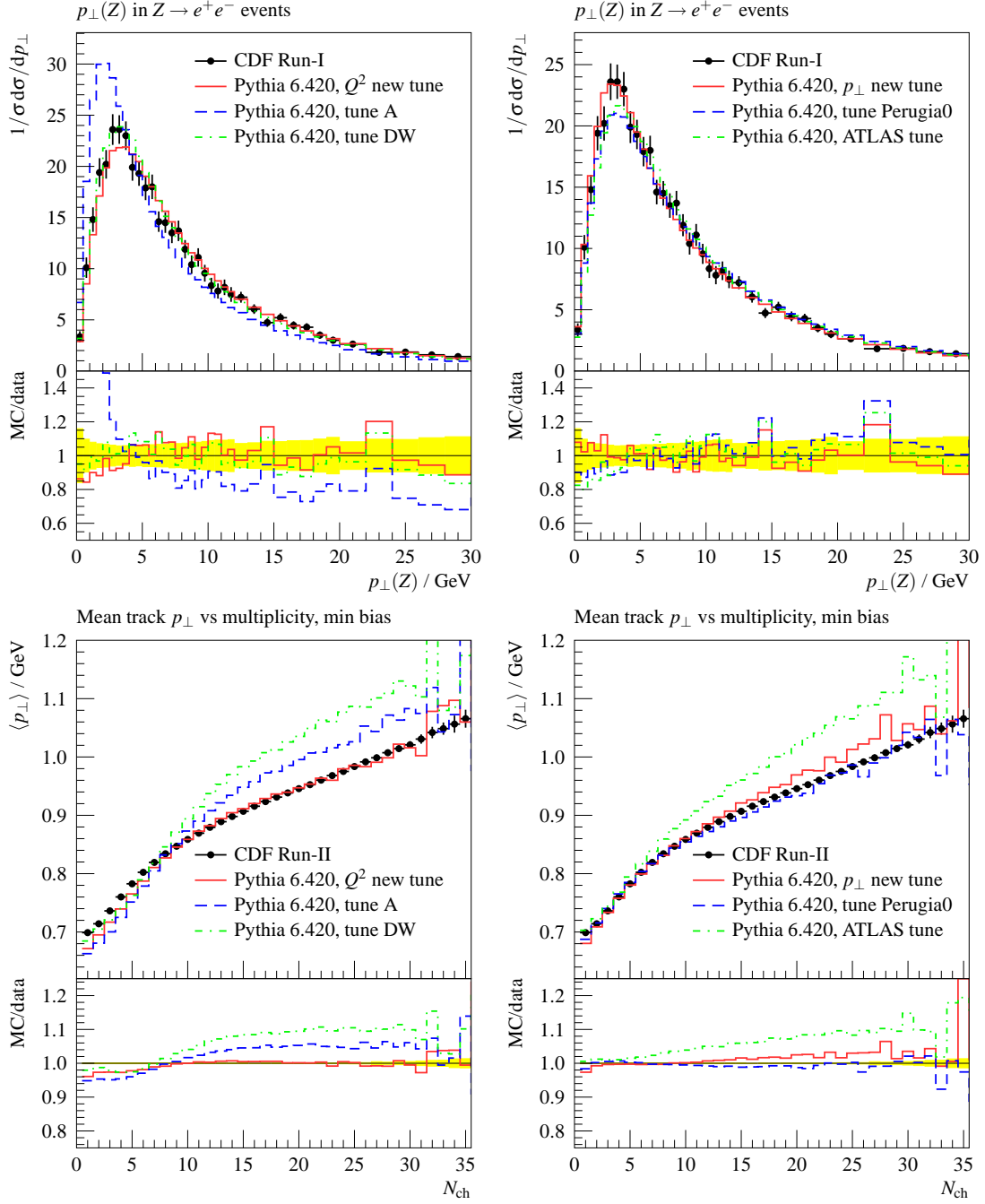


Fig. 3: The upper plots show the Z p_{\perp} distribution as measured by CDF [12] compared to different tunes of the virtuality-ordered shower with the old MPI model (left) and the p_{\perp} -ordered shower with the interleaved MPI model (right). Except for tune A all tunes describe this observable, and also the fixed version of tune A, called AW, is basically identical to DW. The lower plots show the average track p_{\perp} as function of the charged multiplicity in minimum bias events [15]. This observable is quite sensitive to colour reconnection. Only the recent tunes hit the data here (except for ATLAS).

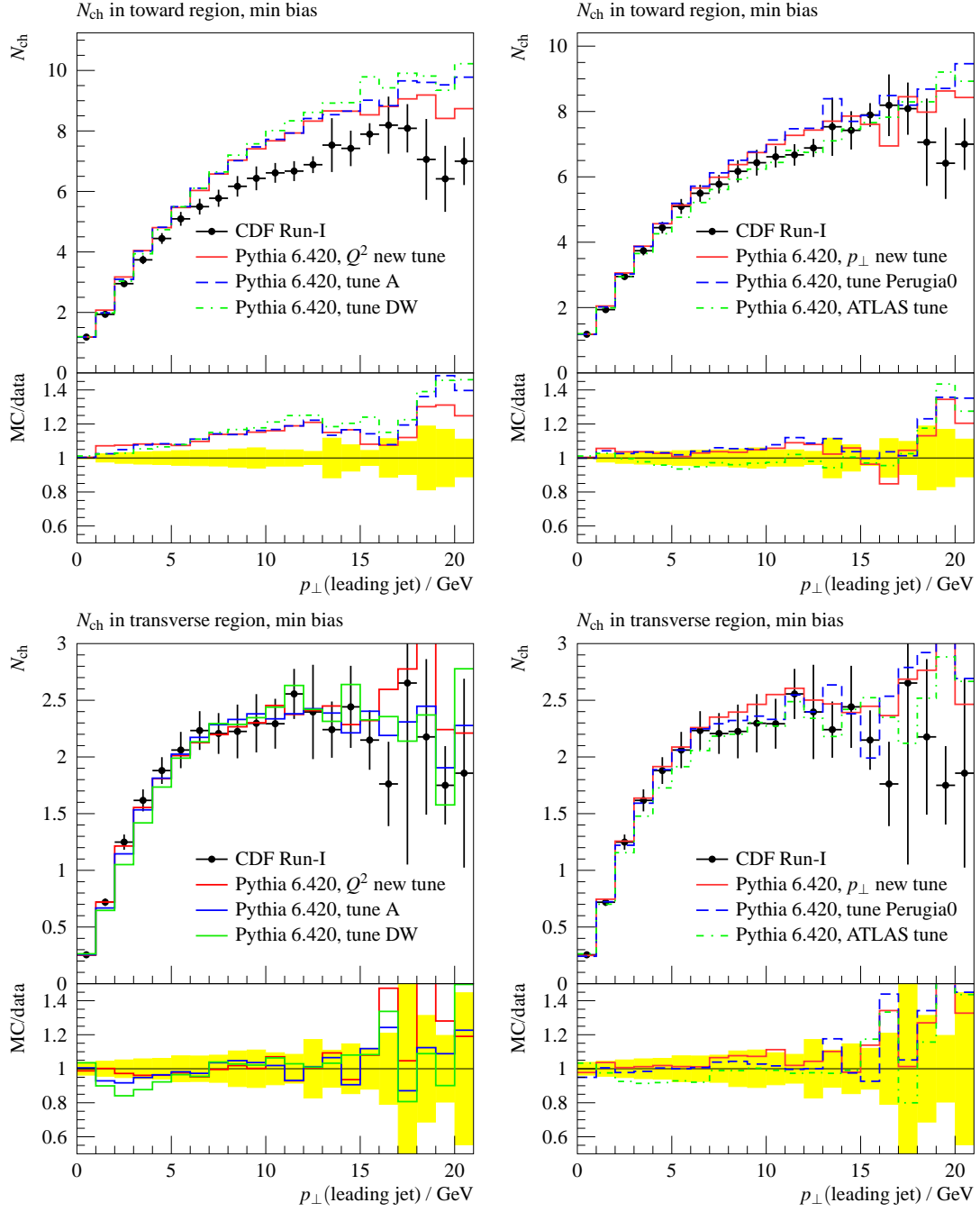


Fig. 4: These plots show the average charged multiplicity in the toward and transverse regions as function of the leading jet p_{\perp} in minimum bias events [13]. On the left side tunes of the virtuality-ordered shower with the old MPI model are shown, while on the right side the p_{\perp} -ordered shower with the interleaved MPI model is used. The old model is known to be a bit too “jetty” in the toward region, which can be seen in the first plot. Other than this, all tunes are very similar.

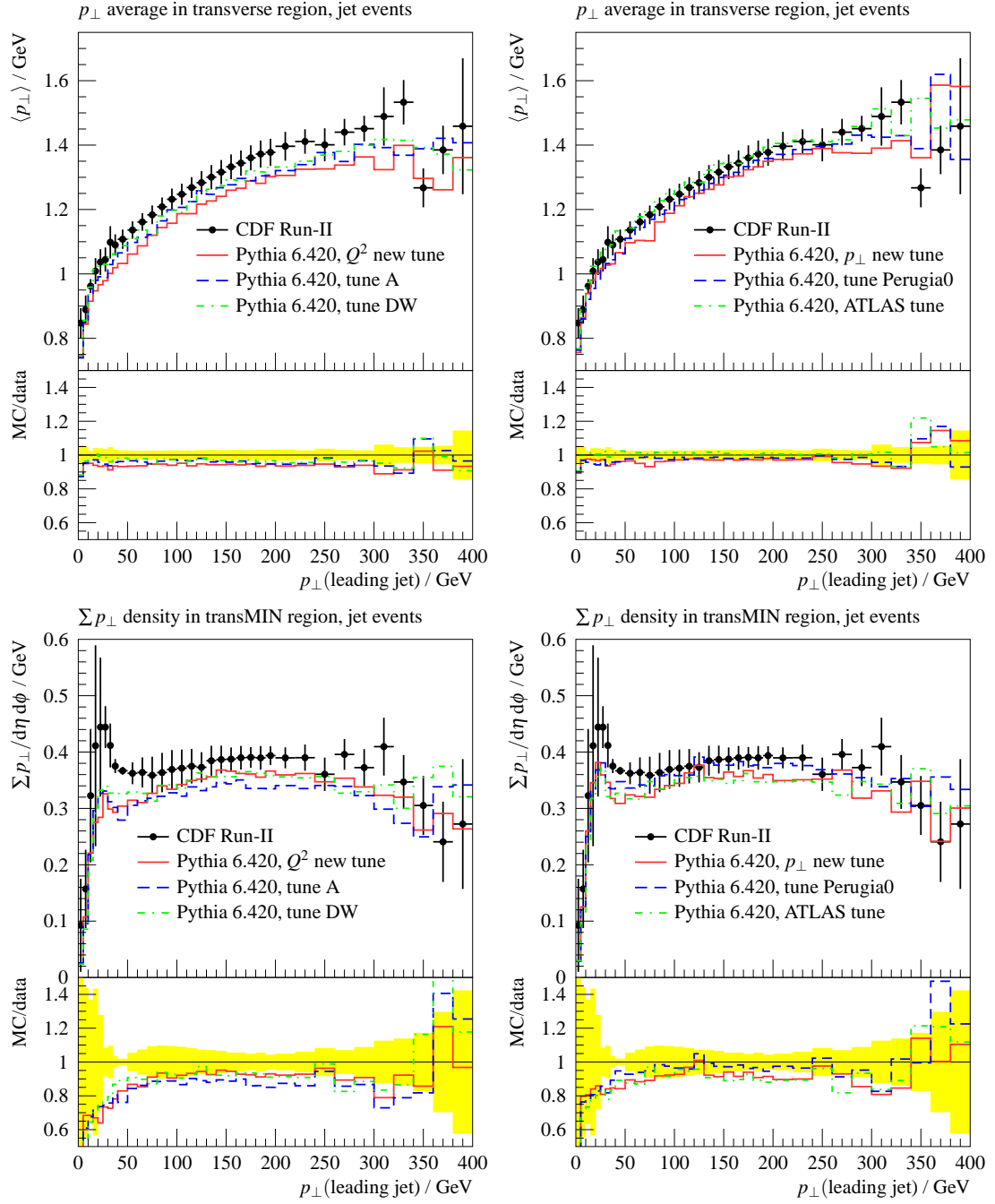


Fig. 5: These plots show the average track p_{\perp} in the transverse region (top) and the Σp_{\perp} density in the transMIN region (bottom) in leading jet events [17]. The new model (on the right) seems do have a slight advantage over the virtuality-ordered shower with the old MPI model shown on the left, both in the turn-on hump and in overall activity.

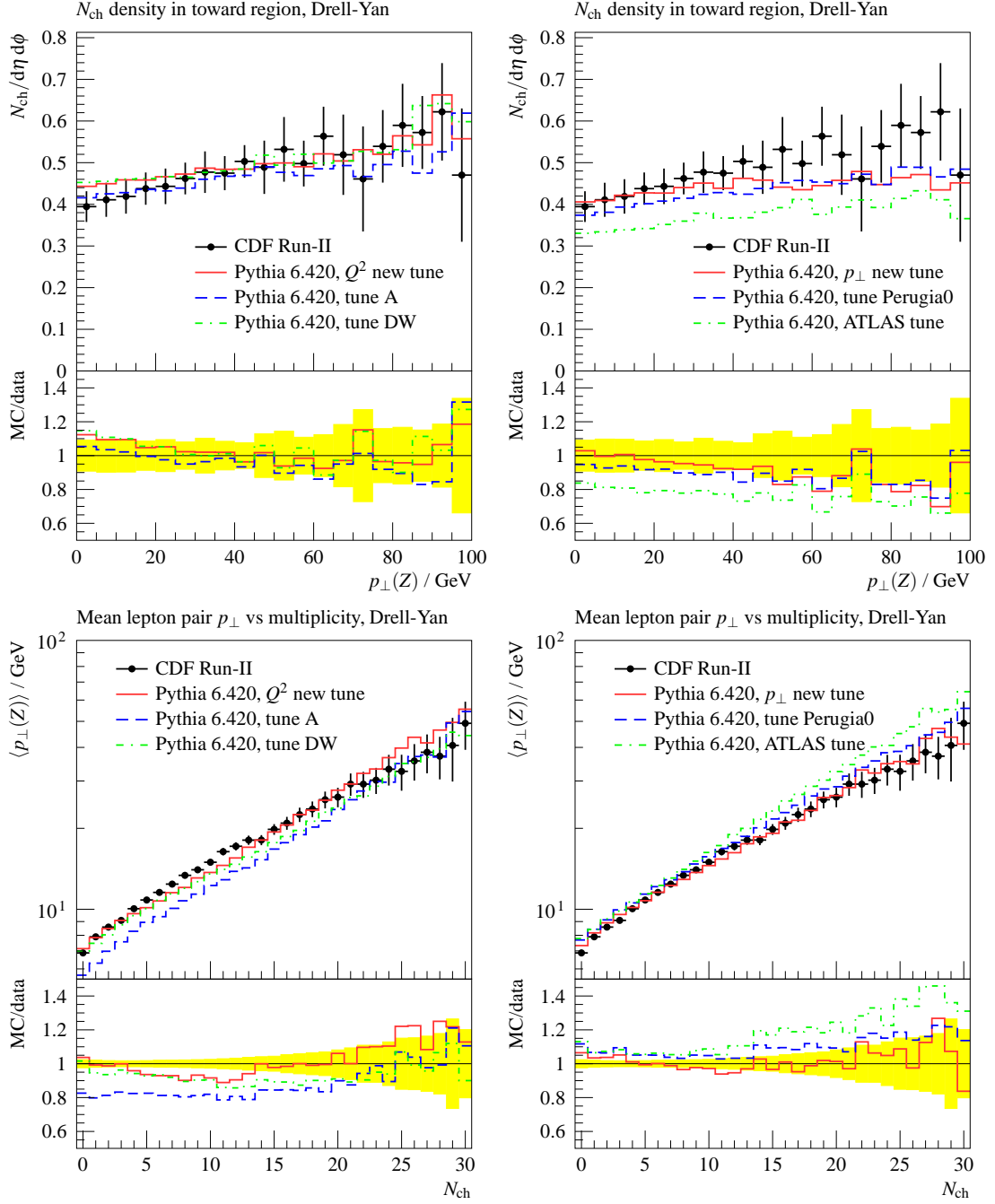


Fig. 6: In Drell-Yan [16] the new MPI model consistently underestimates the activity of the underlying event. Nevertheless, most of the recent tunes are able to describe the multiplicity dependence of the Z p_{\perp} .

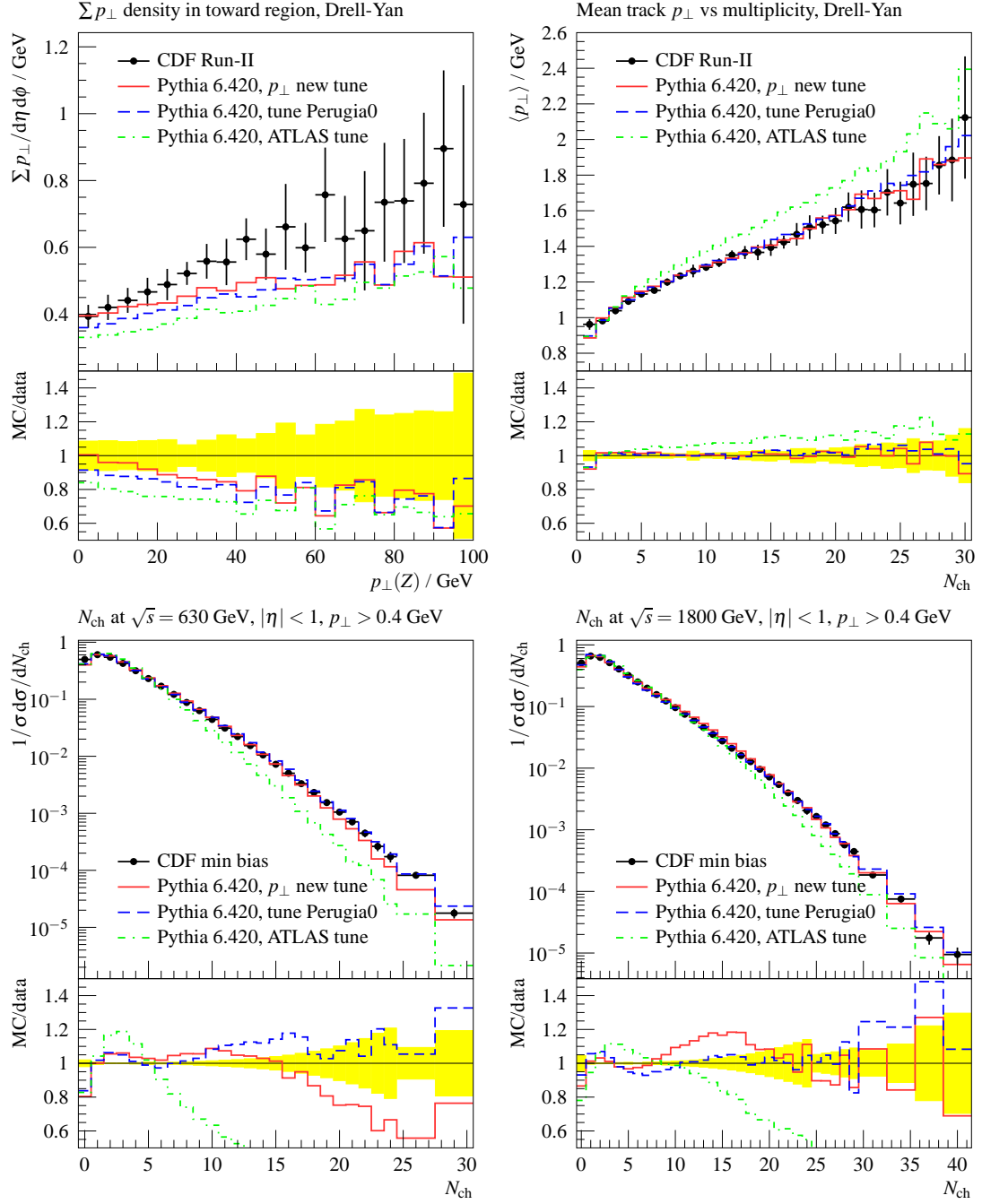


Fig. 7: Some more plots showing the behaviour of the interleaved MPI model and the p_{\perp} -ordered shower. The two upper plots focus on the underlying event in Drell-Yan [16]. On the left we see again that the new model underestimates the activity in Drell-Yan events (like in Fig. 6). Regardless of that, the top right plot shows that the average track p_{\perp} as function of the charged multiplicity is described well – except by the ATLAS tune. The ATLAS tune also has a big problem with the multiplicity distribution in minimum bias events shown in the lower two plots [14]. Even at the reference energy of 1800 GeV this tune fails to match the data.

Parameter	Pythia 6.418 default	Final tune	
PARP(62)	1.0	2.9	ISR cut-off
PARP(64)	1.0	0.14	ISR scale factor for α_S
PARP(67)	4.0	2.65	max. virtuality
PARP(82)	2.0	1.9	p_\perp^0 at reference E_{cm}
PARP(83)	0.5	0.83	matter distribution
PARP(84)	0.4	0.6	matter distribution
PARP(85)	0.9	0.86	colour connection
PARP(86)	1.0	0.93	colour connection
PARP(90)	0.2	0.22	p_\perp^0 energy evolution
PARP(91)	2.0	2.1	intrinsic k_\perp
PARP(93)	5.0	5.0	intrinsic k_\perp cut-off

Table 3: Tuned parameters for the underlying event using the virtuality-ordered shower

Parameter	Pythia 6.418 default	Final tune	
PARP(64)	1.0	1.3	ISR scale factor for α_S
PARP(71)	4.0	2.0	max. virtuality (non-s-channel)
PARP(78)	0.03	0.17	colour reconnection in FSR
PARP(79)	2.0	1.18	beam remnant x enhancement
PARP(80)	0.1	0.01	beam remnant breakup suppression
PARP(82)	2.0	1.85	p_\perp^0 at reference E_{cm}
PARP(83)	1.8	1.8	matter distribution
PARP(90)	0.16	0.22	p_\perp^0 energy evolution
PARP(91)	2.0	2.0	intrinsic k_\perp
PARP(93)	5.0	7.0	intrinsic k_\perp cut-off

Switch	Value	Effect
MSTJ(41)	12	switch on p_\perp -ordered shower
MSTP(51)	7	use CTEQ5L
MSTP(52)	1	use internal PDF set
MSTP(70)	2	model for smooth p_\perp^0
MSTP(72)	0	FSR model
MSTP(81)	21	turn on multiple interactions (new model)
MSTP(82)	5	model of hadronic matter overlap
MSTP(88)	0	quark junctions \rightarrow diquark/Baryon model
MSTP(95)	6	colour reconnection

Table 4: Tuned parameters (upper table) and switches (lower table) for the underlying event using the p_\perp -ordered shower.

4 Conclusions

The Rivet and Professor tools are in a state where they can be used for real tunings and the tuning of Pythia 6.4 has been a significant success. At and around the Perugia workshop a bunch of new tunes appeared on the market: Our Professor tunes, Peter Skand's Perugia tunes (which are based on our flavour and fragmentation parameters), and combinations of the well established Rick Field tunes with our new flavour and fragmentation settings which even improve the agreement with data at the Tevatron. All these tunes are directly available through the PYTUNE routine in Pythia 6.420 or later.

We strongly encourage the LHC experiments to use one of these tunings instead of spending their valuable time on trying to tune themselves. Monte Carlo tuning requires a sound understanding of the models and of the data, and a very close collaboration with the generator authors. In the current situation we highly recommend the use of either Peter Skands' Perugia tune or our new tune if the user wants to go for the new MPI model, or a tune like DWpro or our tune of the virtuality-ordered shower for a more conservative user who wants to use a well-proven model.

Acknowledgements

We want to thank the organisers of the MPI@LHC workshop for this very productive and enjoyable meeting, and we are looking forward to its continuation. Our work was supported in part by the MCnet European Union Marie Curie Research Training Network, which provided funding for collaboration meetings and attendance at research workshops such as MPI@LHC'08. Andy Buckley has been principally supported by a Special Project Grant from the UK Science & Technology Funding Council. Hendrik Hoeth acknowledges a MCnet postdoctoral fellowship.

References

- [1] T. Sjöstrand, S. Mrenna, and P. Skands, JHEP **05**, 026 (2006). hep-ph/0603175.
- [2] K. Hamacher and M. Weierstall (1995). hep-ex/9511011.
- [3] Delphi Collaboration, P. Abreu *et al.*, Z. Phys. **C73**, 11 (1996).
- [4] A. E. Albert, *Regression and the Moore-Penrose pseudoinverse*. Academic Press, New York., 1972.
- [5] K. M. Brown, *Computer oriented methods for fitting tabular data in the linear and nonlinear least squares sense*, in *AFIPS '72 (Fall, part II): Proceedings of the December 5-7, 1972, fall joint computer conference, part II*, pp. 1309–1315. ACM, New York, NY, USA, 1972.
- [6] A. Buckley, H. Hoeth, H. Schulz, and J. E. von Seggern (2009). 0902.4403.
- [7] A. Buckley *et al.* (2006). hep-ph/0605048.
- [8] Particle Data Group Collaboration, C. Amsler *et al.*, Phys. Lett. **B667**, 1 (2008).
- [9] DELPHI Collaboration, G. Barker *et al.* (2002). DELPHI 2002-069 CONF 603.
- [10] OPAL Collaboration, K. Ackerstaff *et al.*, Eur. Phys. J. **C7**, 369 (1999). hep-ex/9807004.
- [11] ATLAS Collaboration, A. Moraes (2009). ATL-PHYS-PROC-2009-045.
- [12] CDF Collaboration, T. Affolder *et al.*, Phys. Rev. Lett. **84**, 845 (2000). hep-ex/0001021.
- [13] CDF Collaboration, T. Affolder *et al.*, Phys. Rev. **D65**, 092002 (2002).
- [14] CDF Collaboration, D. Acosta *et al.*, Phys. Rev. **D65**, 072005 (2002).
- [15] CDF Collaboration, T. Aaltonen *et al.* (2009). 0904.1098.
- [16] CDF Collaboration, D. Kar and R. Field (2008). CDF Note 9351.

- [17] R. Field, *The Underlying Event and Comparisons with MC* (unpublished). These proceedings.
- [18] D0 Collaboration, V. M. Abazov *et al.*, Phys. Rev. Lett. **94**, 221801 (2005). [hep-ex/0409040](#).
- [19] R. Field, “*Minimum Bias*” *Collisions at CDF and CMS*. Talk presented at the CMS Meeting on Min-Bias Monte-Carlo Tunes, CERN, June 28, 2006.
- [20] M. Sandhoff and P. Skands. Presented at Les Houches Workshop on Physics at TeV Colliders, Les Houches, France, 2–20 May 2005, in [hep-ph/0604120](#).
- [21] P. Skands and D. Wicke, Eur. Phys. J. **C52**, 133 (2007). [hep-ph/0703081](#).
- [22] P. Skands, *The “Perugia” Tunes* (unpublished). These proceedings.
- [23] R. Field, *Min-Bias and the Underlying Event at the Tevatron and the LHC*. Talk presented at the Fermilab ME/MC Tuning Workshop, October 4, 2002.
- [24] *1st Joint Workshop on Energy Scaling of Hadron Collisions: Theory/RHIC/Tevatron/LHC*, April 2009. <http://theory.fnal.gov/trtles/>.