Event Generator Tuning

Hendrik Hoeth

Overview

Motivation	–	“why” and “what’s the problem”
Strategy	–	“how to tune”
Examples	–	“reality check”
Plans	–	“what we learned” and “what’s 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: q_0^2, σ_q, Lund A and B, flavour ratios, . . .

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.

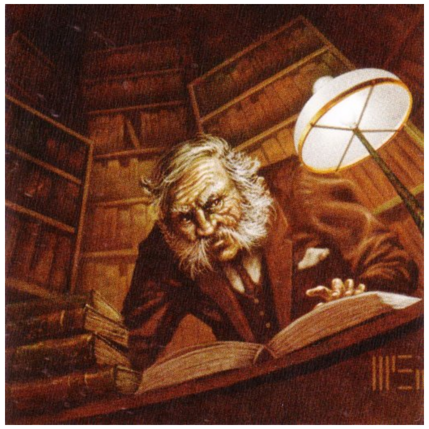
3. Fit this prediction to data (minimal χ^2).

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) 11-60)

Professor and Rivet



Tools for MC tuning have been implemented as an effort within MCnet.

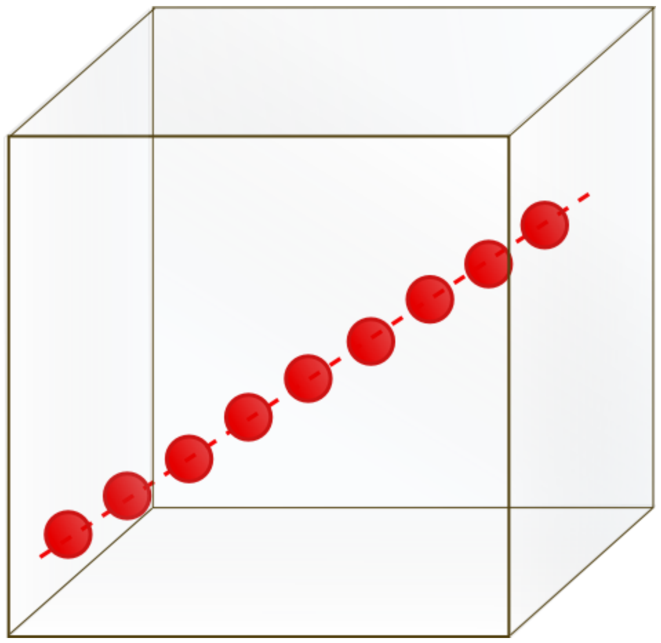
Rivet/Rivetgun: A general tool to steer different MC generators in a common way and to run analyses on generator level. Lots of published analyses are implemented, direct data/MC comparison is very easy. (http://projects.hepforge.org/rivet/)

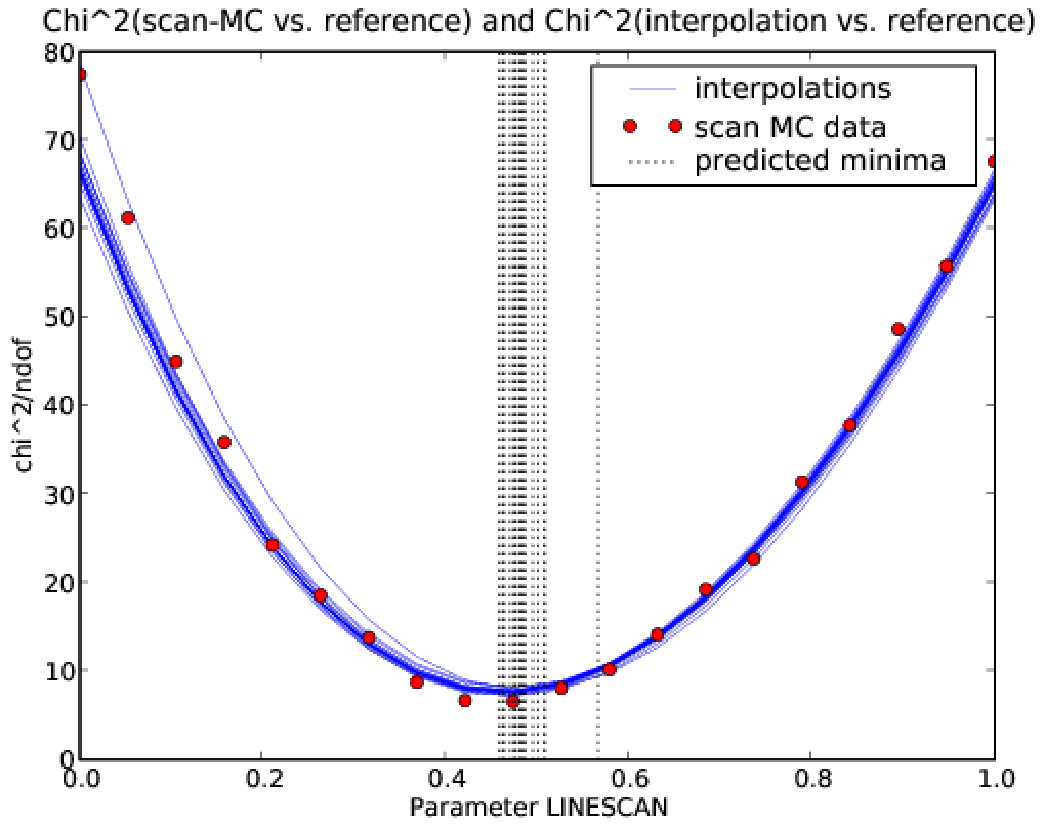
Professor: Implementation of the tuning procedure. Uses Rivet to fill histograms. (http://projects.hepforge.org/professor/)

Reality Check

Machinery is running and has been checked (event generation, interpolation, minimisation).

Example: χ^2 scan through a minimum in 3-dim parameter space:





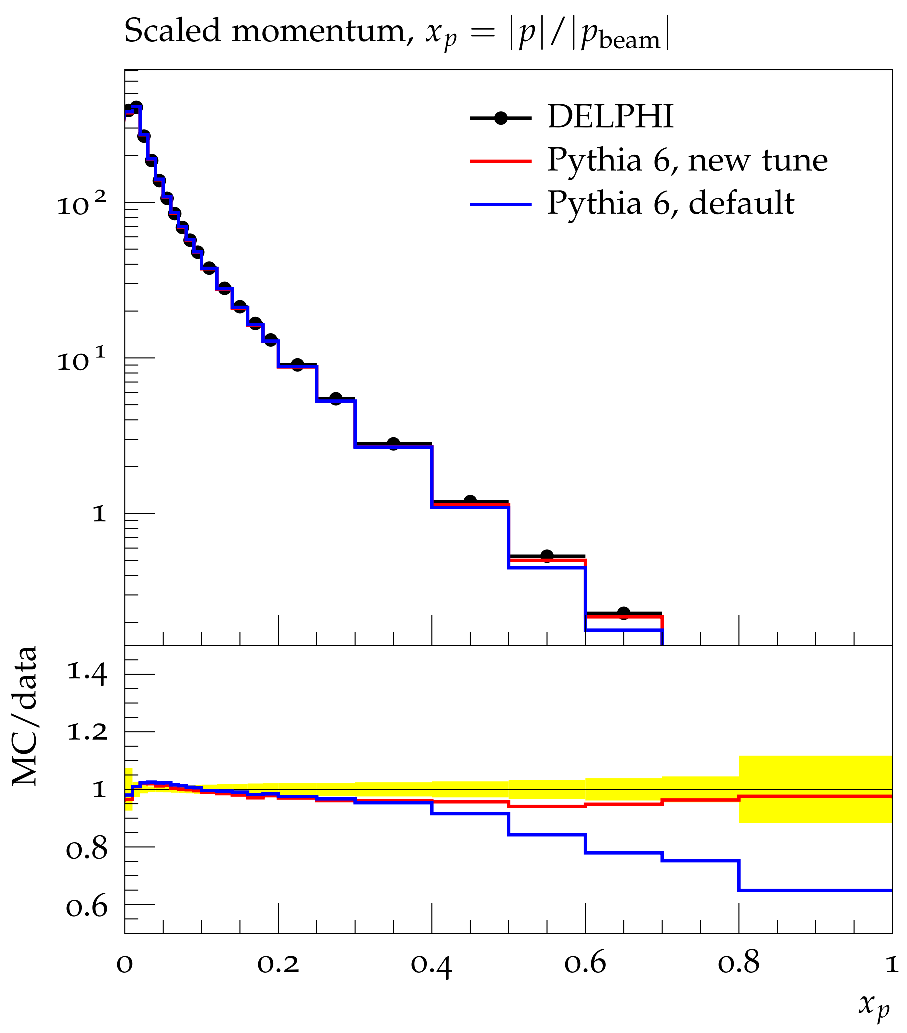
Tuning Pythia 6 – LEP

Two-stage tune of Pythia 6 to LEP data:

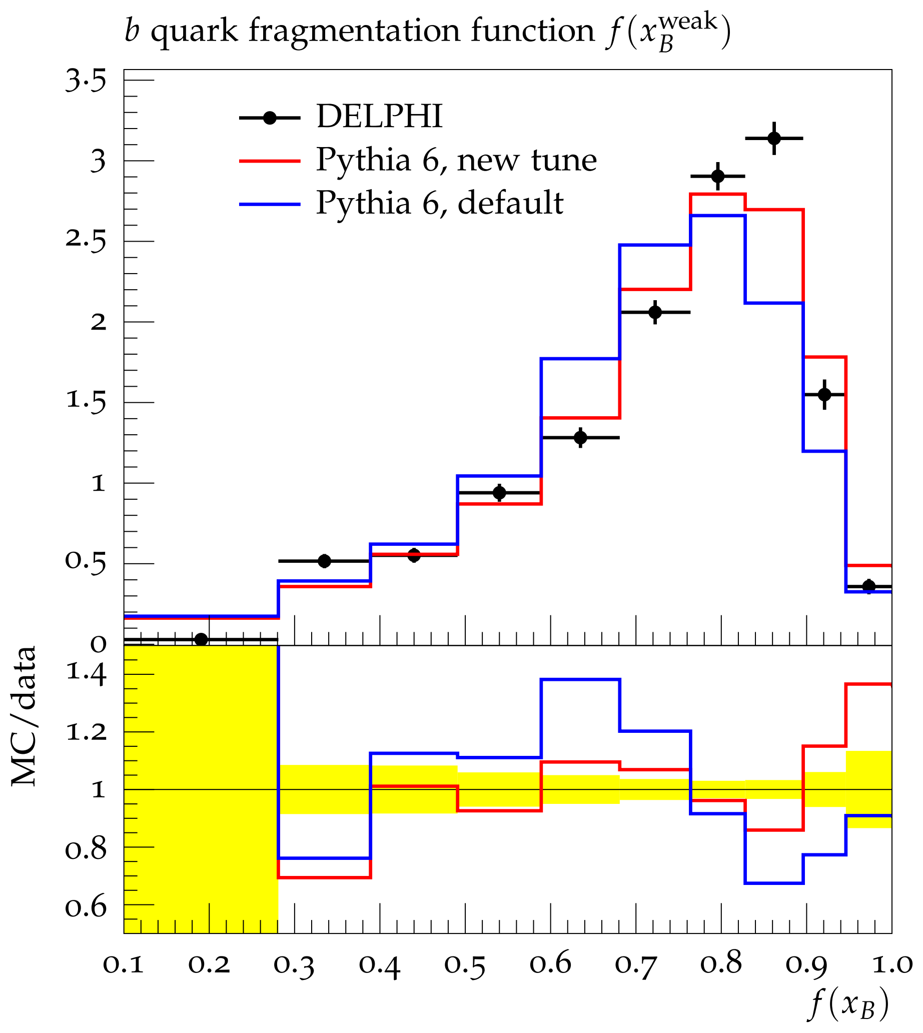
- Flavour parameters. Tuned to identified particle multiplicities, normalized to pions.
- Fragmentation, hadronization. Tuned to event shapes, b-fragmentation measurement, multiplicities, momentum spectra.

Improvement of many particle multiplicities, general improvement of event shapes.

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



Scaled momentum, $x_p = |p|/|p_{\text{beam}}|$
DELPHI
Pythia 6, new tune
Pythia 6, default
10^2
10^1
1
MC/data
1.4
1.2
1
0.8
0.6
0
0.2
0.4
0.6
0.8
1
x_p



b quark fragmentation function $f(x_B^{\text{weak}})$
DELPHI
Pythia 6, new tune
Pythia 6, default
3.5
3
2.5
2
1.5
1
0.5
0
MC/data
1.4
1.2
1
0.8
0.6
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
$f(x_B)$

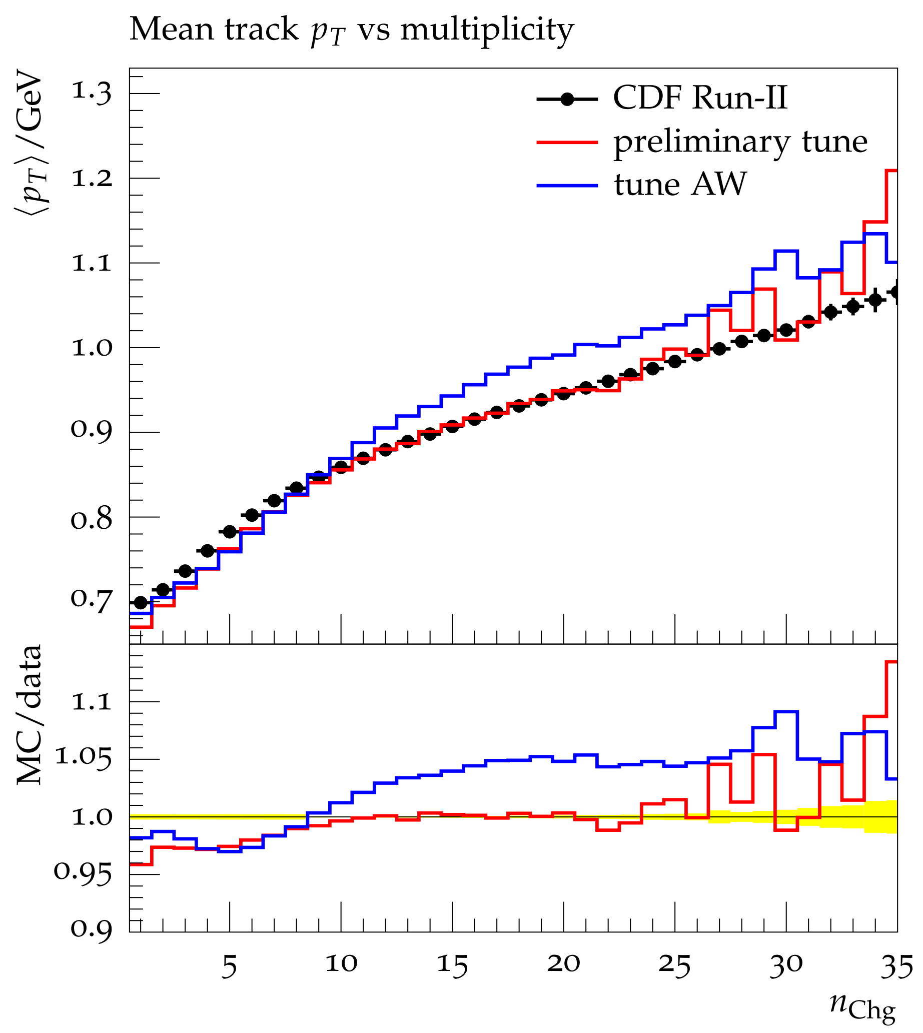
Tuning Pythia 6 – Tevatron

Currently we are tuning the underlying event parameters to Tevatron data ($>$ 50 distributions from CDF and DØ).

First results are very promising – in terms of overall χ^2 for all our distributions we already beat tune A/AW.

Work in progress – preliminary results will be presented at the MPI@LHC workshop in two weeks.

Just to give you a taste . . .



Plans

Some of the things we are working on or have in mind for the near future:

- Finish Pythia 6 UE
- p_T ordered shower in Pythia 6
- Pythia 8
- Sherpa cluster hadronisation
- Herwig++

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 in MCnet.

Tunings of Pythia and Sherpa are in progress, first results will be published soon.

Final plea to the experimentalists

Tunings are only as good as the input data.

We need your acceptance corrected data to improve your Monte Carlo.

Backup Slides

1. Choosing parameters

Pick the parameters you want to tune:

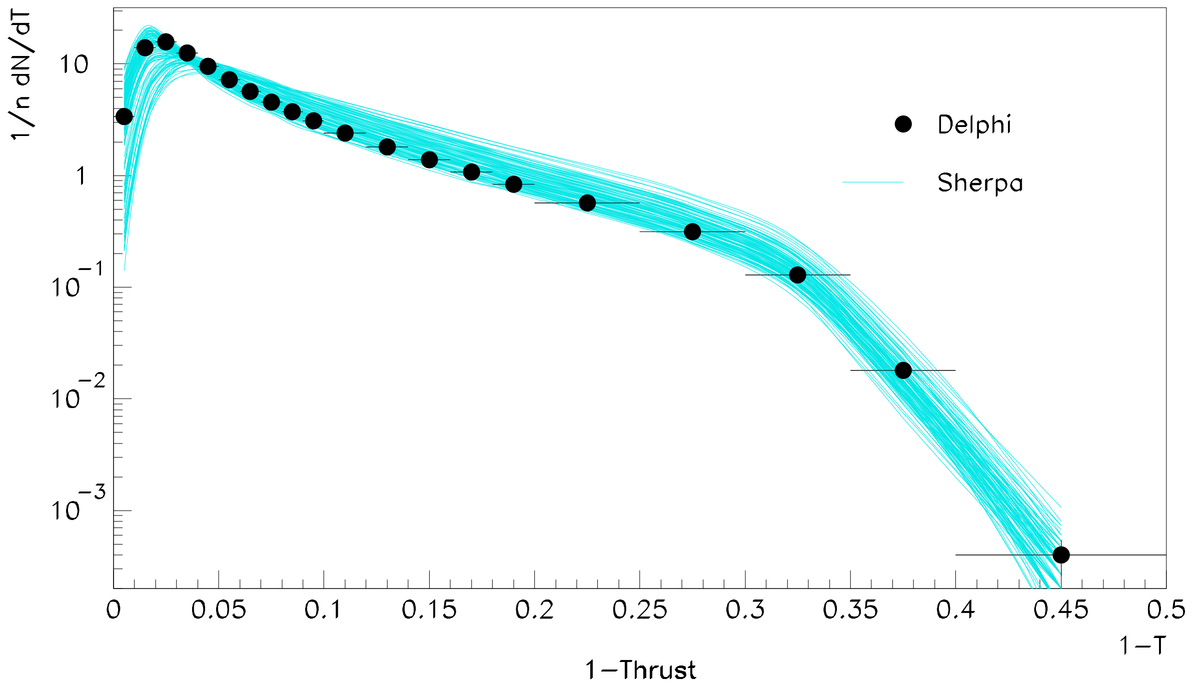
- Tune everything that is important.
- But remember: Each additional parameter adds one dimension to the parameter space.

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 setting.

2. Predict the Monte Carlo

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

Using a second order polynomial takes the correlations between the parameters into account:

$$X_{\mathrm{MC}}(p_1, p_2, \ldots, p_n) =$$

$$A_0 + \sum_{i=1}^{n} B_i p_i + \sum_{i=1}^{n} C_i p_i^2 + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} D_{ij} p_i p_j + \cdots$$

3. Fit the prediction to data

Having A_0, B_i, C_i and D_{ij} we can predict the MC response for any set of parameters very fast. This prediction can be fitted to data, minimising the χ^2:

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

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).