

# 对撞机蒙特卡洛模拟： Parton Shower

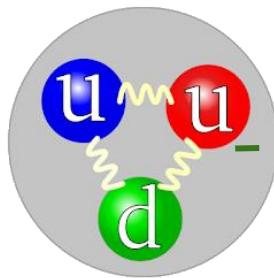
## 2021对撞机唯像学暑期学校



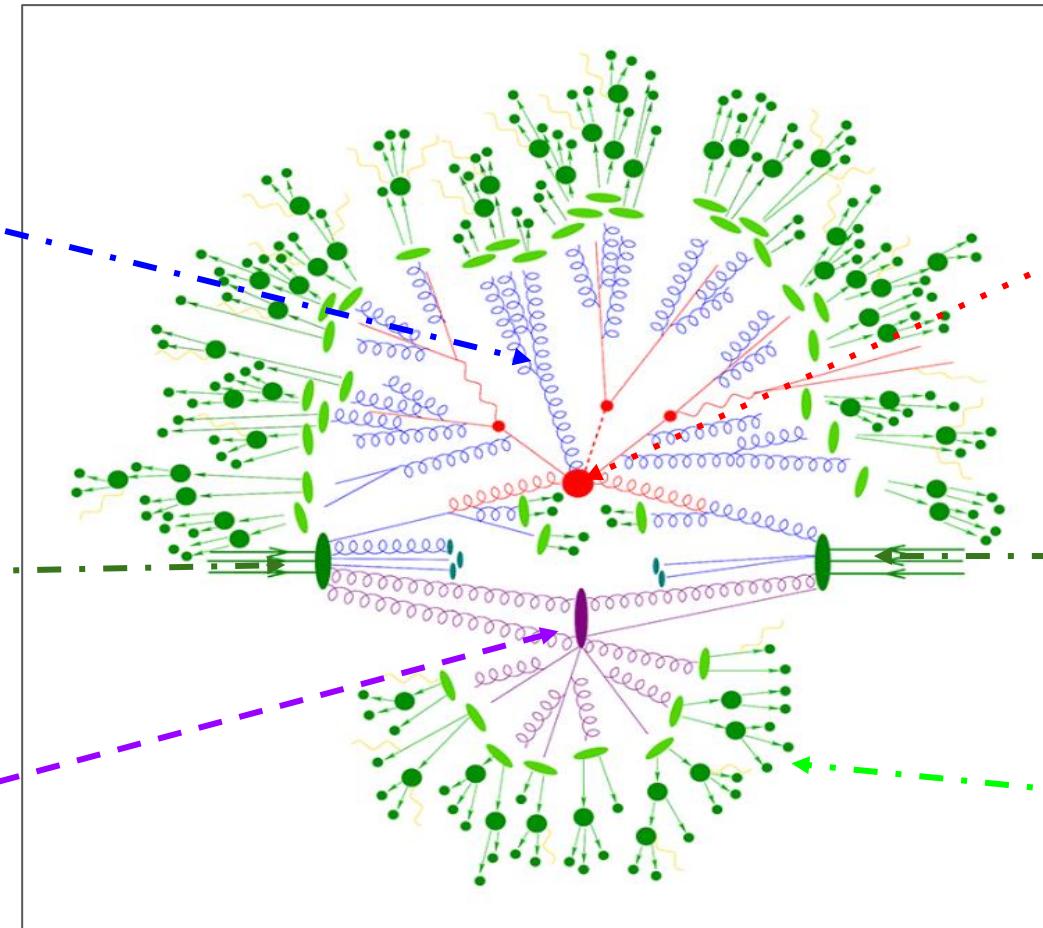
李强 北京大学物理学院技术物理系  
[qliphy0@pku.edu.cn](mailto:qliphy0@pku.edu.cn)

# 高能对撞

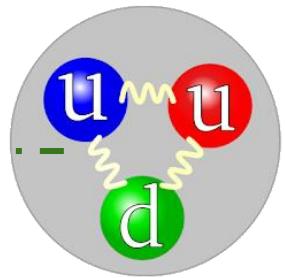
QCD演化:  
Parton Shower



多重散射



硬散射



强子化

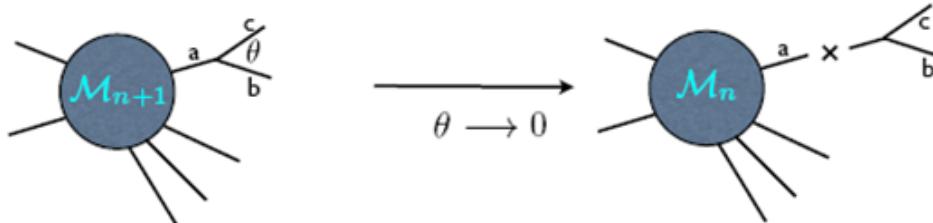
# Outline

- QCD and Parton Shower
- Pythia, Herwig, Sherpa
- Pythia6 example
- Pythia8
  - Event display
  - Drell Yan
  - MadGraph -> Pythia8: LO
  - MadGraph -> Pythia8: Matching
  - MadGraph -> Pythia8: NLO, FxFx
  - Hepmc
  - Status Code: H decay
  - Status Code: ZGamma vs ZJets
- Rivet

[https://conference.ippp.dur.ac.uk/event/309/contributions/1286/attachments/1054/1203/MC3\\_krauss.pdf](https://conference.ippp.dur.ac.uk/event/309/contributions/1286/attachments/1054/1203/MC3_krauss.pdf)

<https://indico.cern.ch/event/368497/contributions/1787017/attachments/1134046/1621975/gieseke-2.pdf>

# QCD and Parton Shower



## DGLAP splitting function

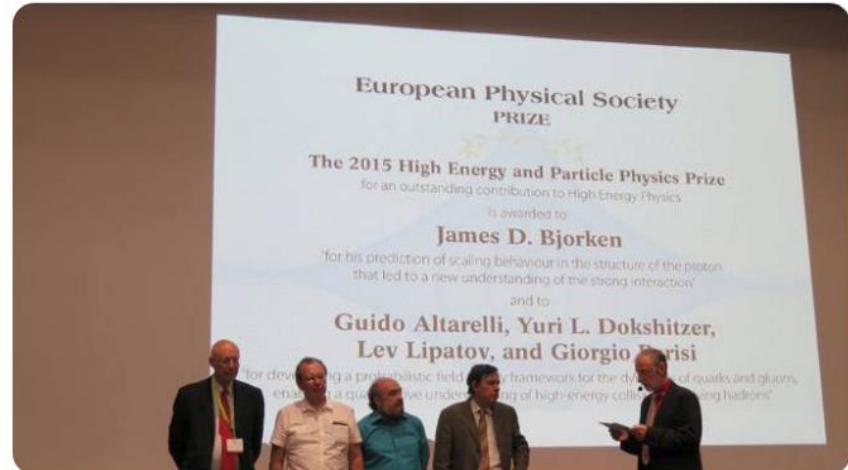
$$dP_{a \rightarrow bc} = \frac{\alpha_s}{2\pi} \frac{dQ^2}{Q^2} P_{a \rightarrow bc}(z) dz$$

where  $P_{q \rightarrow qg} = \frac{4}{3} \frac{1+z^2}{1-z}$ ,

$$P_{g \rightarrow gg} = 3 \frac{(1-z(1-z))^2}{z(1-z)},$$

$$P_{g \rightarrow q\bar{q}} = \frac{n_f}{2} (z^2 + (1-z)^2) \quad (n_f = \text{no. of quark flavours})$$

EPS HEP prize 2015 a Bjorken, Altarelli, Dokshitzer, Lipatov e Parisi  
#EPSHEP2015

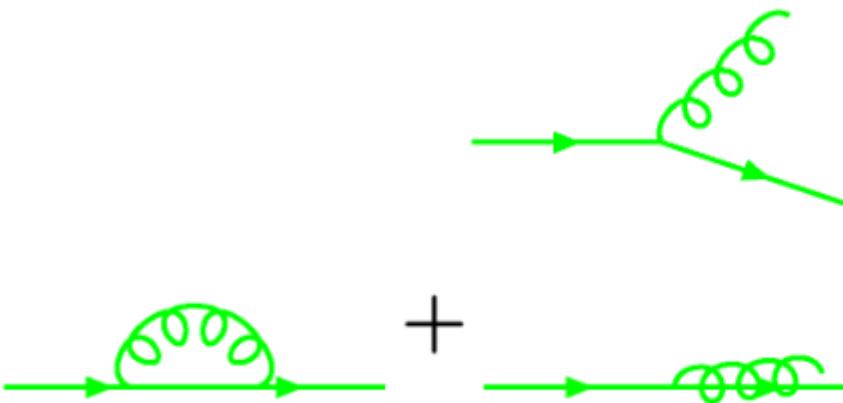


Collinear limit: Universal

Q is ordering parameter, can be virtuality, either the transverse momentum PT, or angle between products.

# Sudakov Factor

Collinear parton pair → single parton  
resolution criterion, eg  $K_\perp > Q_0$

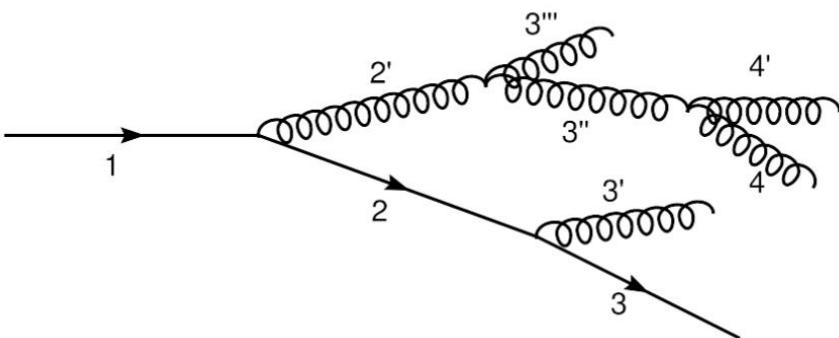


Resolvable emission:  
Finite

Virtual + Unresolvable  
emission: Finite

Unitarity:  $P(\text{resolved}) + P(\text{unresolved}) = 1$

# Sudakov Factor



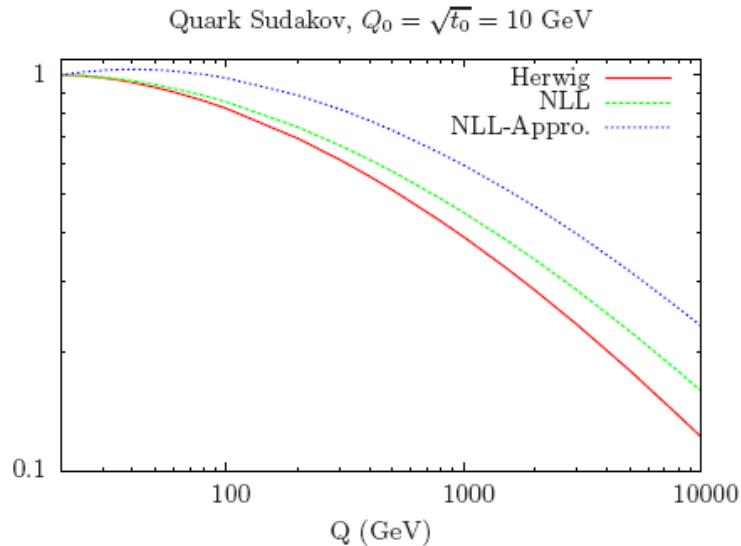
Probability that particle  $a$  does not emit between scales  $Q^2$  and  $t$ :

$$\Delta(Q^2, t) = \prod_k \left[ 1 - \sum_{bc} \frac{dt_k}{t_k} \int dz \frac{d\phi}{2\pi} \frac{\alpha_s}{2\pi} P_{a \rightarrow bc}(z) \right] =$$

$$\exp \left[ - \sum_{bc} \int_t^{Q^2} \frac{dt'}{t'} dz \frac{d\phi}{2\pi} \frac{\alpha_s}{2\pi} P_{a \rightarrow bc}(z) \right] = \exp \left[ - \int_t^{Q^2} dp(t') \right].$$

- $\Delta(Q^2, t)$  is the Sudakov form factor.
- Property:  $\Delta(A, B) = \Delta(A, C)\Delta(C, B)$ .

**Sudakov Factor**



e.g. a TeV quark has large probability to split

# Sudakov Factor

SOVIET PHYSICS JETP

VOLUME 3, NUMBER 1

AUGUST, 1956

## Vertex Parts at Very High Energies in Quantum Electrodynamics

V. V. SUDAKOV

(Submitted to JETP editor Nov. 4, 1954)

J. Exper. Theoret. Phys. USSR 30, 87-95 (January 1956)

A method is developed for calculating Feynman integrals with logarithmic accuracy, working to any order of perturbation theory. The method is applied to calculate the vertex part in quantum electrodynamics for a certain range of values of the momenta. The result is displayed as the sum of a perturbation series.

THE technique of Feynman<sup>1</sup> for calculating matrix elements in quantum electrodynamics is only suitable for the lowest-order approximations, since the algebraic complexities increase extremely rapidly when we consider contributions to the matrix element from higher-order perturbations. When perturbation theory is not applicable and it is necessary to consider the sum of the entire perturbation series\*, another technique must be developed. For example, one elegant method<sup>3</sup> of calculating integrals with logarithmic accuracy depends on changing  $k_0$  into  $ik_0$ . This method is, however, not applicable to all cases. In particular, it is inapplicable to the calculation of the vertex part  $\Gamma_\sigma(p, q; l)$  in

the case when the absolute value of the square of one of the vectors  $p, q, l$  is much larger than the absolute squares of the other two vectors. This case is especially important for concrete physical applications. There appear in this case terms with the structure  $e^2 L_1 L_2$ , a product of two big logarithms entering with each power of  $e^2$  (we call these doubly-logarithmic terms). But the earlier method<sup>3</sup> can give only terms with the structure  $e^2 L$  (singly-logarithmic terms), in which one large logarithm enters with each power of  $e^2$ .

1. To explain the method<sup>4</sup> of obtaining the doubly-logarithmic terms, we shall consider as an example the integral

The number of relevant diagrams in the  $(2n)$ 'th order is equal to  $(n!)$  (the number of permutations  $i_1, i_2, \dots, i_n$ ). We can now sum the contributions from these diagrams over all values of  $n$ , and obtain

(31)

$$\begin{aligned}\Gamma_\sigma(p, q; l) &= \gamma_\sigma \sum_{n=1}^{\infty} \frac{1}{n!} \left( -\frac{e^2}{2\pi} \ln \left| \frac{l^2}{p^2} \right| \ln \left| \frac{l^2}{q^2} \right| \right)^n \\ &= \gamma_\sigma \exp \left\{ -\frac{e^2}{2\pi} \ln \left| \frac{l^2}{p^2} \right| \ln \left| \frac{l^2}{q^2} \right| \right\}.\end{aligned}$$

Vladimir Sudakov

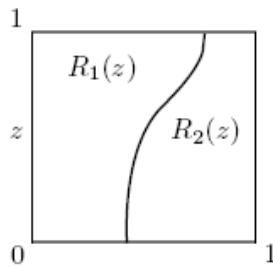
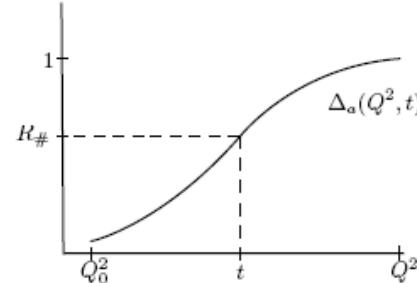


# Parton Shower in brief

## Implementation

- ▶ Extract the evolution variable  $t$  of the branching by solving the equation  $\Delta(Q^2, t) = R_\#,$  with  $R_\#$  a flat random number between 0 and 1.

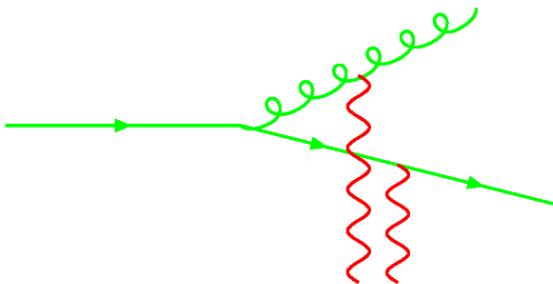
This correctly reproduces the probability distribution since the probability of extracting a splitting scale  $t$  between  $t_1$  and  $t_2$  is  $\Delta(Q^2, t_2) - \Delta(Q^2, t_1).$



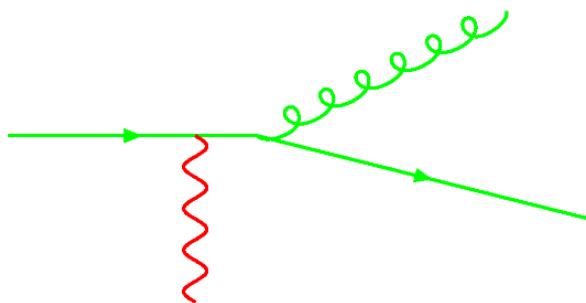
- ▶ Extract the energy sharing  $z$  and the daughter identities  $b$  and  $c$  according to  $P_{a \rightarrow bc}(z).$  For two possible branchings  $P_1(z)$  and  $P_2(z)$  one can call  $R_i(z) = P_i(z)/(P_1(z) + P_2(z)),$  and choose  $z$  and parton identities by extracting a random point in the plane.

- ▶ Extract  $\phi$  (flat).
- ▶ Reiterate (updating the maximum scale for the Sudakov) until all the 'external' partons are characterized by a scale smaller than a threshold  $Q_0^2 \sim 1 \text{ GeV}.$
- ▶ Put partons on shell and hadronize.

# Angular ordering

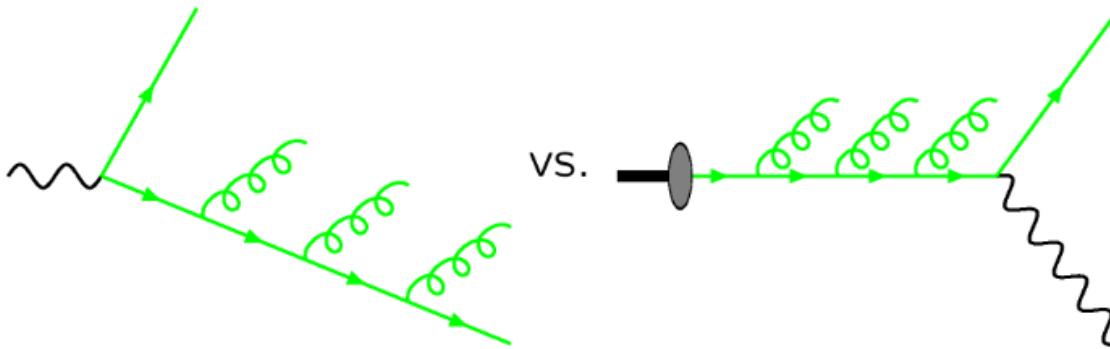


soft gluon comes from everywhere in event  
→ Quantum interference.  
→ Spoils independent evolution picture?



outside angular ordered cones, soft gluons sum coherently: only see colour charge of whole jet.  
**Soft gluon effects fully incorporated by using  $\theta$  as evolution variable:  
angular ordering**

# Space/time-like shower



Final State radiation:  
Time-like shower

Initial State radiation:  
Space-like shower

In principle identical to final state.  
In practice different because both ends of evolution fixed

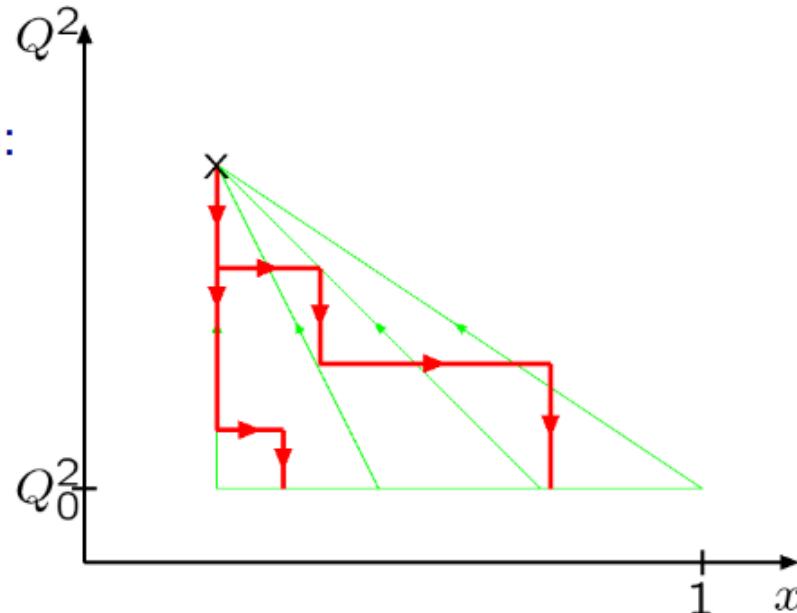
# Backward Evolution

DGLAP evolution: pdfs at  $(x, Q^2)$  as function of pdfs at  $(> x, Q_0^2)$

Evolution paths sum over all possible events.

Formulate as backward evolution:  
start from hard scattering and work down in  $q^2$ , up in  $x$  towards incoming hadron.

Algorithm identical to final state with  $\Delta_i(Q^2, q^2)$  replaced by  $\Delta_i(Q^2, q^2)/f_i(x, q^2)$ .



# Reshuffling

After shower: original partons acquire virtualities  $q_i^2$

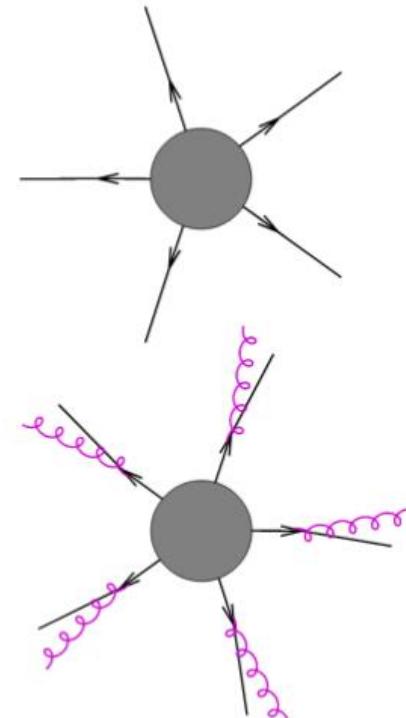
→ boost/rescale jets:  
Started with

$$\sqrt{s} = \sum_{i=1}^n \sqrt{m_i^2 + \vec{p}_i^2}$$

we *rescale* momenta with common factor  $k$ ,

$$\sqrt{s} = \sum_{i=1}^n \sqrt{q_i^2 + k\vec{p}_i^2}$$

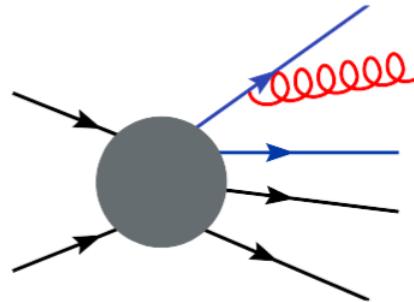
to preserve overall energy/momentum.  
→ resulting jets are boosted accordingly.



# Dipole-recoil vs Global-recoil

Exact kinematics when recoil is taken by spectator(s).

- Dipole showers.
- Ariadne.
- Recoils in Pythia.



By default the recoil of an ISR emission is taken by the whole final state. The option below gives an alternative approach with local recoils, where only one final-state parton takes the recoil of an emission.

**flag SpaceShower:dipoleRecoil (default = off)**  
Option to switch on the dipole-recoil scheme as described above.

# Starting Scale

“power showers” vs “wimpy showers”

TimeShower:pTmaxMatch = 0/1/2

SpaceShower:pTmaxMatch = 0/1/2

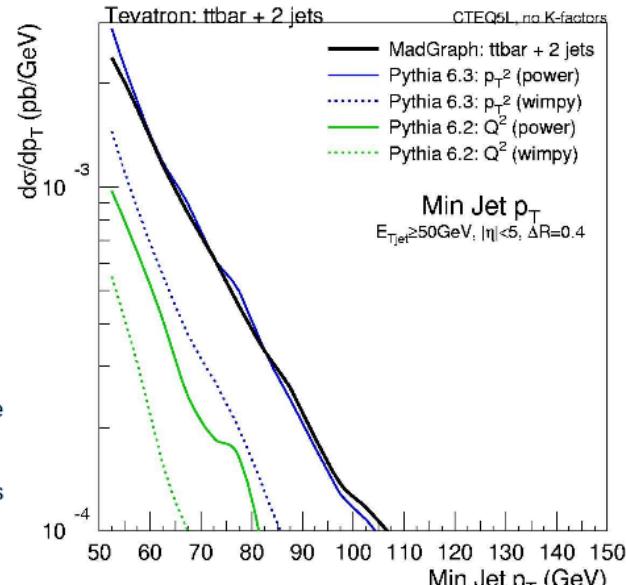
```
mode TimeShower:pTmaxMatch (default = 1; minimum = 0; maximum = 2)
```

Way in which the maximum shower evolution scale is set to match the scale of the hard process itself.

**option 0** : (i) if the final state of the hard process (not counting subsequent resonance decays) contains at least one quark (*u, d, s, c, b*), gluon or photon then *pT\_max* is chosen to be the factorization scale for internal processes and the *scale* value for Les Houches input; (ii) if not, emissions are allowed to go all the way up to the kinematical limit (i.e. to half the dipole mass). This option agrees with the corresponding one for *spacelike showers*. There the reasoning is that in the former set of processes the ISR emission of yet another quark, gluon or photon could lead to double-counting, while no such danger exists in the latter case. The argument is less compelling for timelike showers, but could be a reasonable starting point.

**option 1** : always use the factorization scale for an internal process and the *scale* value for Les Houches input, i.e. the lower value. This should avoid double-counting, but may leave out some emissions that ought to have been simulated. (Also known as wimpy showers.)

**option 2** : always allow emissions up to the kinematical limit (i.e. to half the dipole mass). This will simulate all possible event topologies, but may lead to double-counting. (Also known as power showers.)



<http://home.thep.lu.se/~torbjorn/pythia81html/TimelikeShowers.html>  
<http://home.thep.lu.se/~torbjorn/pythia81html/SpacelikeShowers.html>

# Matrix Element Correction

TimeShower:MEcorrections = on/off

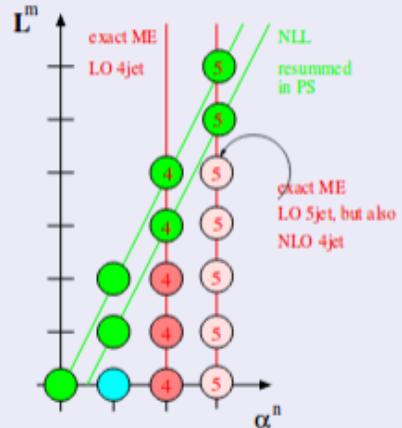
SpaceShower:MEcorrections = on/off

Matrix-element corrections have been implemented, which correct the first jet emission to the full LO matrix-element.

## ME vs. PS

- Matrix elements good for:  
hard, large-angle emissions;  
take care of interferences.
- Parton shower good for:  
soft, collinear emissions;  
resums large logarithms.
- Want to combine both!  
Avoid double-counting.

## Orders in ME & PS



MLM, NLO, FxFx, each has different Pythia shower setting!

# Overview



**PYTHIA** (successor to JETSET, begun in 1978)  
originated in string hadronization studies.  
Historically strong interest in soft physics: MPI, CR.  
**Angantyr** model for pA/AA: Leif Lönnblad next.



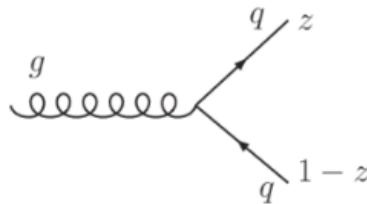
**Herwig** (successor to EARWIG, begun in 1984)  
originated with coherent showers (angular ordering).  
MPI, CR and cluster hadronization added.  
Only simple event stacking for pA/AA.



**Sherpa** (APACIC++/AMEGIC++, begun in 2000)  
originated with matrix elements calculations.  
Emphasis on (N)NLO match & merge, less on soft.  
Heavy-ion effort under way (JEWEL, SHRIMPSS, ...).

# Overview

Generator	Recoil	Evolution
HERWIG++	Global	$\tilde{q}^2$
HERWIG++	Dipole	$p_{\perp \text{dip}}^2$
HERWIG++	Dipole	$q_{\text{dip}}^2$
PYTHIA8	Dipole	$p_{\perp}^2$
VINCIA	Antenna	$p_{\perp \text{ant}}^2$
VINCIA	Antenna	$m_{\text{ant}}^2$

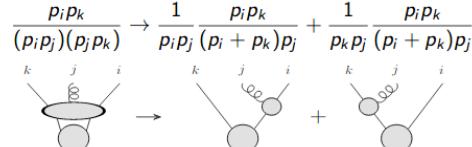


Herwig7

$$\begin{aligned}\tilde{q}^2 &= \frac{q_i^2 - m_i^2}{z(1-z)}; \\ &= \frac{p_T^2 + (1-z)m_j^2 + zm_k^2 - z(1-z)m_i^2}{z^2(1-z)^2}; \\ &= \frac{2q_j \cdot q_k + m_j^2 + m_k^2 - m_i^2}{z(1-z)};\end{aligned}$$

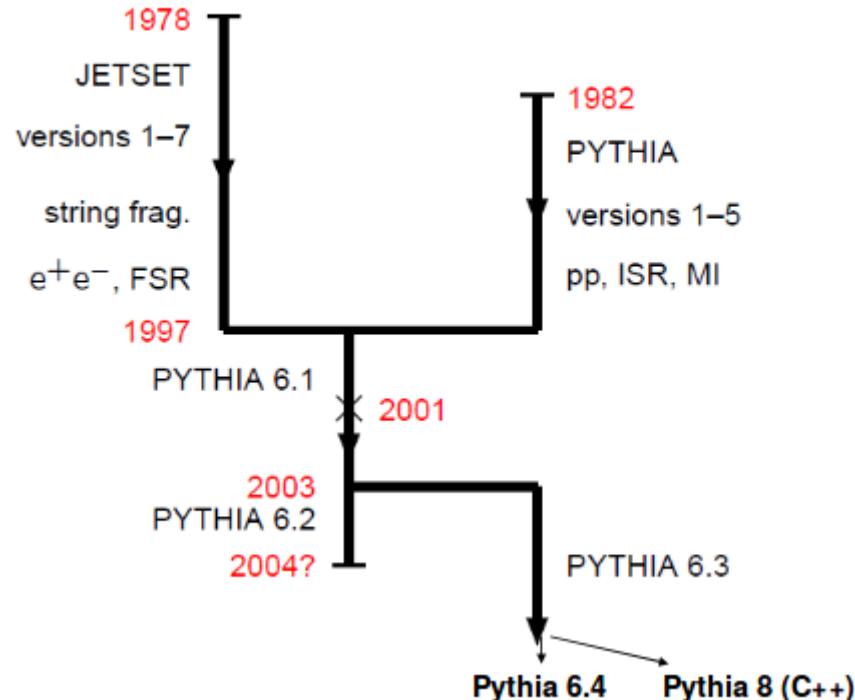
PS	Splitting function	Recoil scheme	Construction
PYTHIA8	$1 \rightarrow 2$	Local	DGLAP
HERWIG++ (angular)	$1 \rightarrow 2$	Global	DGLAP
HERWIG++ (dipole)	$1 \rightarrow 2$	Local	CS Dipoles
SHERPA/CSSHOWEE++	$1 \rightarrow 2$	Local	CS Dipoles
ARIADNE	$2 \rightarrow 3$	Local	Antenna
VINCIA	$2 \rightarrow 3$	Local	Antenna
SHERPA/ANTS	$2 \rightarrow 3$	Local	Antenna
KRKMC	$1 \rightarrow 2$	Global	DGLAP
DEDUCTOR	$n \rightarrow n+1$	Local	NS subtraction

combination of parton and dipole shower picture  
 $\rightarrow$  partial fractioning soft eikonal      [Catani,Seymour Nucl.Phys.B485\(1997\)291](#)



# Pythia

## PYTHIA history



Core Pythia program is small and self-contained: ~160k lines code, ~20 MB gzipped tarball.

Quick & easy to install, well documented and many examples, download from  
<http://home.thep.lu.se/Pythia/>

# Herwig7

## Herwig Evolution

HERWIG



HERWIG (Hadron Emission Reactions With Interfering Gluons):

Fortran code, last version 6.521  
(1992-2002)

[Marchesini, Webber, Abbiendi, Corcella, Knowles, Moretti, Odagiri, Richardson, Seymour, Stanco]

Herwig++ (C++ rewrite, 2004):

[Bähr, Gieseke, Gigg, Grellscheid, Hamilton, Latunde-Dada, Plätzer, Richardson, Seymour, Sherstnev, Tully, Webber]  
last version 2.7.1 (2014)

[Bellm, Gieseke, Grellscheid, Papafistathiou, Platzer, Richardson, Rohr, Schuh, Seymour, AS, Wilcock, Zimmermann]  
intended to fully replace Fortran version

experimental and phenomenological evolution over time  
⇒ precision as key goal

Herwig++ 3.0 → **Herwig 7.0** Evolution of fHERWIG/Herwig++

subsumed as "7 > 6.5". "Better than fHERWIG in any aspect plus more".

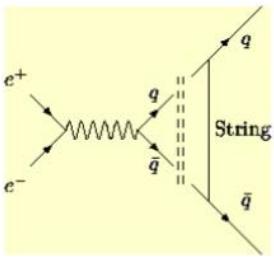
[Bellm, Gieseke, Grellscheid, Plätzer, Rauch, Reuschle, Richardson, Schichtel, Seymour, AS, Wilcock, Fischer, Harrendorf, Nail, Papafistathiou, D. Rauch]

# Sherpa

User Inputs	Matrix Elements	Parton Showers	Soft Physics	Interfaces/Outputs
<b>Initial Beams</b> <ul style="list-style-type: none"> <li>• collider setup</li> <li>• PDFs (built-in, LHAPDF)</li> <li>• beam spectra</li> </ul> <b>Parameters/Models</b> <ul style="list-style-type: none"> <li>• FeynRules/UFO</li> <li>• couplings</li> <li>• masses</li> <li>• variations</li> <li>• shower settings</li> <li>• non-perturbative parameters</li> </ul> <b>Physics Process</b> <ul style="list-style-type: none"> <li>• parton level</li> <li>• perturbative order (QCD/EW)</li> <li>• selectors</li> <li>• matching/merging</li> <li>• partonic decays</li> </ul> 	<b>Matrix Element Generators</b> <ul style="list-style-type: none"> <li>• AMEGIC</li> <li>• COMIX</li> <li>• CS subtraction</li> </ul> <b>1-loop Amplitudes</b> <ul style="list-style-type: none"> <li>• OpenLoops</li> <li>• Recola</li> <li>• GoSam</li> <li>• BLHA</li> </ul> 	<b>CS-Shower (default)</b> <ul style="list-style-type: none"> <li>• dipole shower</li> <li>• fully massive</li> <li>• QED splittings</li> </ul> <b>DIRE</b> <ul style="list-style-type: none"> <li>• hybrid dipole-parton shower algorithm</li> <li>• fully massive</li> </ul> 	<b>Hadronisation</b> <ul style="list-style-type: none"> <li>• AHADIC: a cluster fragmentation model</li> <li>• interface to Pythia string fragmentation</li> </ul> 	<b>Output Formats</b> <ul style="list-style-type: none"> <li>• HepMC</li> <li>• LHEF</li> <li>• Root Ntuple</li> </ul> 
<b>Matching and Merging</b>				
<p><b>Automated MC@NLO style matching</b></p> <p><b>Multijet-merging algorithms</b></p> <ul style="list-style-type: none"> <li>• based on truncated showers</li> <li>• tree-level and one-loop matrix elements: MEPS@LO and MEPS@NLO</li> <li>• approximate electroweak corrections</li> </ul> <p><b>NNLO QCD with parton showers</b></p> <ul style="list-style-type: none"> <li>• selected processes only</li> </ul>		<p><b>Hadron Decays</b></p> <ul style="list-style-type: none"> <li>• decay tables for hadronic resonances</li> <li>• dedicated form-factor models, e.g. <math>\tau</math>, <math>B</math>, <math>\Lambda</math></li> <li>• spin correlations</li> <li>• YFS QED corrections</li> <li>• partonic channels</li> </ul> 	<p><b>Underlying Event</b></p> <ul style="list-style-type: none"> <li>• multiple parton interactions</li> <li>• beam-remnant colours</li> <li>• intrinsic transverse momentum</li> </ul>	<p><b>Interfaces</b></p> <ul style="list-style-type: none"> <li>• RIVET analyses</li> <li>• C++/Python ME access</li> <li>• MCgrid</li> <li>• integration into ATLAS/CMS</li> </ul> 

# Hadronization

## Lund string model (Pythia)



$q\bar{q}$  pair as pointlike source of string.

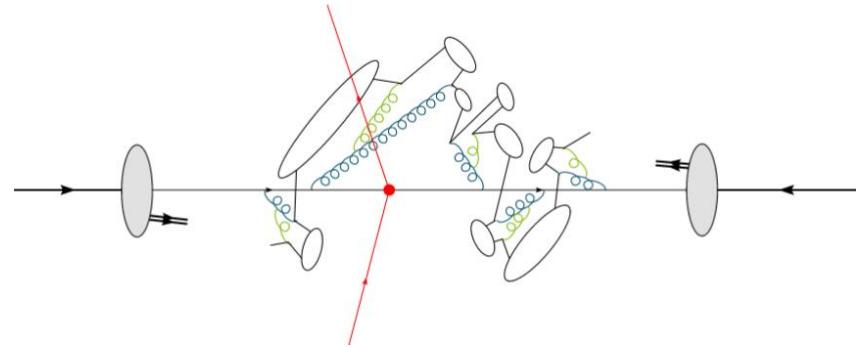
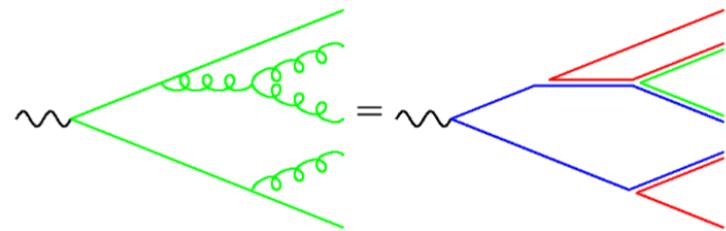
String energy  $\sim$  intense chromomagnetic field  
→ Additional quark pairs created by QM tunneling.

Ajacent breaks form hadrons.

u	d	$\bar{d}$	d	$\bar{d}$	s	$\bar{s}$	d	$\bar{d}$	u	$\bar{u}$	u	$\bar{d}_o$	$ud_o$	s	$\bar{s}$	u
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
p <sup>-</sup>	w	$\bar{K}^{*0}$	K <sup>0</sup>	$\pi^+$	$\bar{p}$	$\Lambda$	$\bar{\Lambda}$	$\bar{K}^+$								
8	7	6	5	4	3	2	1									
1	2	3	4	5	6	7	8									

## Cluster model (Herwig)

Large  $N_C$  limit → planar graphs dominate.  
Gluon = colour — anticolourpair



# Installation

<http://lcgapp.cern.ch/project/simu/HepMC/download/> 2.06.09

```
./configure --with-momentum=GeV --with-length=MM --prefix=/home/qliphy/Desktop/common/hepmc
```

<http://home.thep.lu.se/~torbjorn/Pythia.html>

**HepMC**  
**a C++ Event Record for Monte Carlo Generators**

PY8.2.4.4

```
./configure --with-hepmc2=/home/qliphy/Desktop/common/hepmc --with-root=/home/qliphy/Desktop/common/root/
```

-> May need to adjust Makefile.inc manually

```
# ROOT configuration.
```

```
ROOT_USE=true
```

```
ROOT_BIN=/home/qliphy/Desktop/common/root/bin/
```

```
ROOT_INCLUDE=/home/qliphy/Desktop/common/root/include/
```

```
ROOT_LIB=/home/qliphy/Desktop/common/root/lib/
```

<https://root.cern.ch>



# Installation

<https://launchpad.net/mg5amcnlo>

MG267

install ExRootAnalysis

install Delphes

install lhapdf6

(./bin/lhapdf --source=http://lhapdfsets.web.cern.ch/lhapdfsets/current/ update

./bin/lhapdf --source=http://lhapdfsets.web.cern.ch/lhapdfsets/current/ get NNPDF31\_nnlo\_as\_0118 )

install pythia8

or MG242+pythia6



<https://pythiasix.hepforge.org>

**PYTHIA 6.4 is the last PYTHIA 6 version**, containing the most up-to-date features and bug fixes. It is no longer actively developed but is maintained here in a legacy state. PYTHIA 6.4 was a direct continuation of version 6.3, and version 6.400 contained no news relative to PYTHIA 6.327.

# 1st example: pythia6 Zprime

<https://github.com/qliphy/2020HEPMC/tree/master/pythia6>

gfortran -w -o lq test.f pythia6403.f

```
MSTP(61)=0 ! No IRS
MSTP(71)=0 ! No FRS
MSTP(81)=0 ! No Beam Frag
MSTP(111)=0 ! No Hadronization
ECM=14000.D0
PMAS(32,1)=200.d0
MSTP(44)=7 ! 7 for full gamma, Z, Z' included
CKIN(1)=20.d0
CKIN(2)=-1.0d0
MSTP(32)=4 ! Choice of Q, Q^2

NEV=10000
```

```
DO 2000 IEV=1,NEV
CALL PYEVNT
CALL PYLIST(2) ! List Event information

C      PT=PYP(11,10)
C      CALL PYFILL(1,PT,1.D0) ! Draw PT figure

MMee=-(PYP(11,1)+PYP(12,1))**2-(PYP(11,2)+PYP(12,2))**2
$ -(PYP(11,3)+PYP(12,3))**2+(PYP(11,4)+PYP(12,4))**2

MMee=dsqrt(MMee)

CALL PYFILL(1,MMee,1.D0) ! Draw MMee figure

2000 Continue
```

```
MSTP(44) : (D = 7) treatment of Z0/Z0/γ* interference in matrix elements.
= 1 : only γ* included.
= 2 : only Z0 included.
= 3 : only Z'0 included.
= 4 : only Z0/γ* (with interference) included.
= 5 : only Z'0/γ* (with interference) included.
= 6 : only Z0/Z0 (with interference) included.
= 7 : complete Z0/Z0/γ* structure (with interference) included.
```

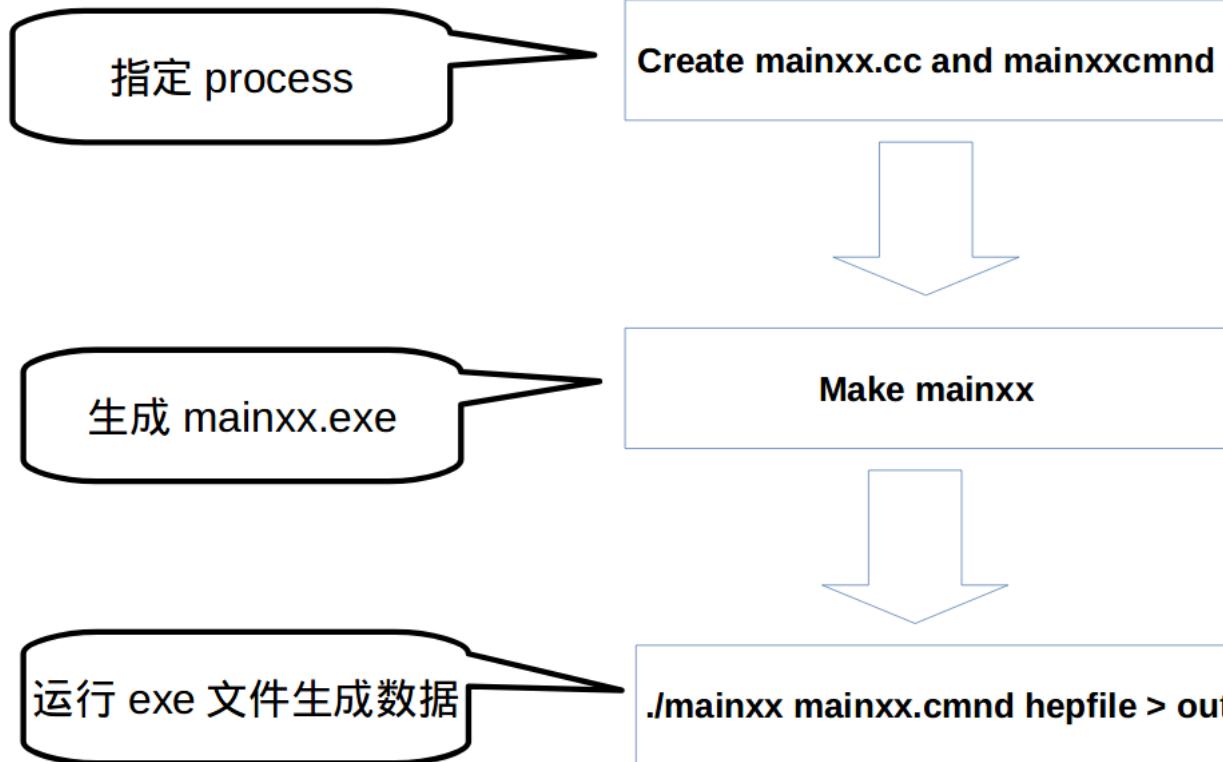
# 1st example: pythia6 Zprime

1 !p+!	21	2212	0	0	0	0.00000	0.00000	6999.99994	7000.00000
0.93827									
<hr/>									
2 !p+!	21	2212	0	0	0	0.00000	0.00000	-6999.99994	7000.00000
0.93827									
<hr/>									
3 !dbar!	21	-1	1	0	0	1.02856	1.42259	4.78690	5.09864
0.00000									
4 !d!	21	1	2	0	0	-1.36139	-0.87034	-402.42233	402.42557
0.00000									
5 !dbar!	21	-1	3	0	0	1.02856	1.42259	4.78690	5.09864
0.00000									
6 !d!	21	1	4	0	0	-1.36139	-0.87034	-402.42233	402.42557
0.00000									
7 !Z'0!	21	32	0	0	0	-0.33283	0.55225	-397.63543	407.52421
89.22797									
8 !e-!	21	11	7	0	0	1.43208	-0.13434	-402.39715	402.39972
0.00051									
9 !e+!	21	-11	7	0	0	-1.76491	0.68659	4.76172	5.12448
0.00051									
<hr/>									
10 (Z'0)	11	32	7	11	12	-0.33283	0.55225	-397.63543	407.52421
89.22797									
11 e-	1	11	8	0	0	1.43208	-0.13434	-402.39715	402.39972
0.00051									
12 e+	1	-11	9	0	0	-1.76491	0.68659	4.76172	5.12448
0.00051									
13 n0	1	2112	1	0	0	-0.72532	-0.82453	3274.02002	3274.02034
0.93957									
14 u	A	2	2	1	0	-0.30324	-0.59806	3721.03527	3721.03534
0.33000									
15 uu_1	V	1	2203	2	0	1.36139	0.87034	-6597.41986	6597.42011
0.77133									

Multi-copy, status code, mothers, daughters

# Pythia8 test

Pythia



# Pythia8 test

## Pythia

```
#include "Pythia8/Pythia.h"
using namespace Pythia8;
int main() {
    Pythia pythia;// Declare Pythia object
    pythia.readString("Top:gg2ttbar = on");//process
    pythia.readString("Beams:eCM = 8000."); //energy
    pythia.init();// Initialize; incoming pp beams is default.
    pythia.next();// Generate an(other) event. Fill event record.
    return 0;
}
```

# 2nd example: pythia8 built-in DY

<http://home.hep.lu.se/~torbjorn/pythia81html/ElectroweakProcesses.html>

Single boson

**flag WeakSingleBoson:all (default = off)**

Common switch for the group of a single  $\gamma^* Z^0$  or  $W^{+-}$  production.

**flag WeakSingleBoson:ffbar2gmZ (default = off)**

Scattering  $f\bar{f} \rightarrow \gamma^* Z^0$ , with full interference between the  $\gamma^*$  and  $Z^0$ . Code 221.

**flag WeakSingleBoson:ffbar2W (default = off)**

Scattering  $f\bar{f} \rightarrow W^{+-}$ . Code 222.

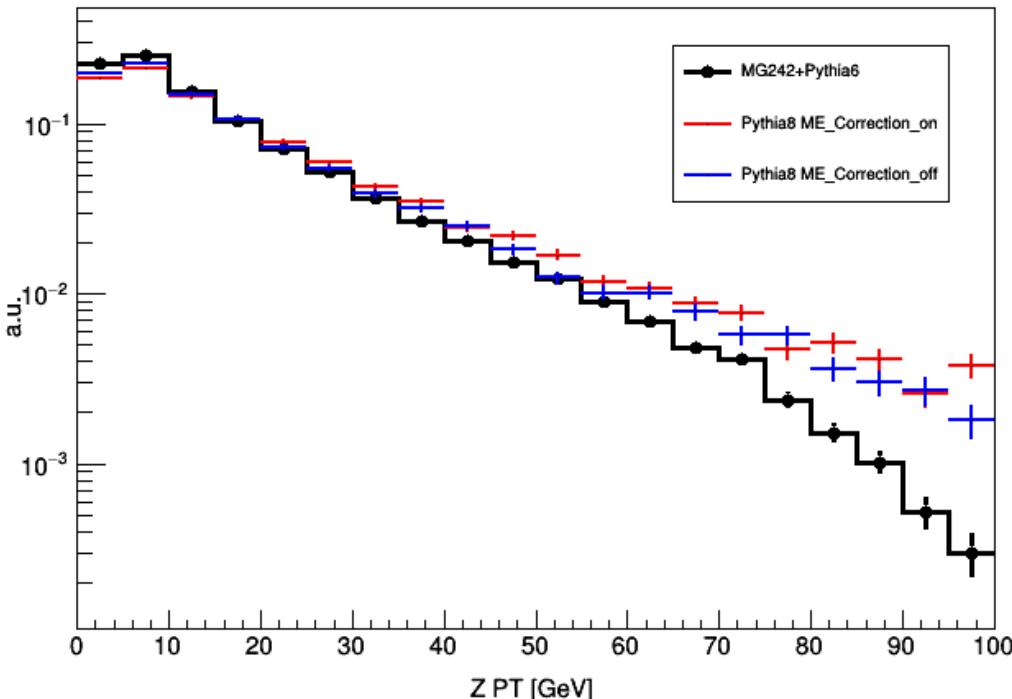
[https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test0\\_main02-DY/main02/main02.cc](https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test0_main02-DY/main02/main02.cc)

```
Pythia pythia;
pythia.readString("Beams:idB = 2212");
pythia.readString("Beams:eCM = 14000.");
pythia.readString("WeakSingleBoson:ffbar2gmZ = on");
pythia.readString("PhaseSpace:mHatMin = 80.");
pythia.readString("PhaseSpace:mHatMax = 120.");
pythia.init();
Hist pTZ("dN/dpTZ", 100, 0., 100.);
```

```
for (int iEvent = 0; iEvent < 1000; ++iEvent) {
    if (!pythia.next()) continue;
    // Loop over particles in event. Find last Z0 copy. Fill its pT.
    int iZ = 0;
    for (int i = 0; i < pythia.event.size(); ++i)
        if (pythia.event[i].id() == 23) iZ = i;
    pTZ.fill( pythia.event[iZ].pT() );
    ptz=pythia.event[iZ].pT();
    ez=pythia.event[iZ].e();
    // End of event loop. Statistics. Histogram. Done.
    t->Fill();
}
```

# 2nd example: pythia8 built-in DY

$pp \rightarrow Z \rightarrow \mu^+\mu^-$  at 13TeV LHC



## MG242+pythia6:

generate p p > z > mu+ mu-  
0.0 = ptl  
-1.0 = etal  
0. = drll  
80.0 = mmll  
120.0 = mmllmax  
MSTP(61)=1  
MSTP(71)=1  
MSTJ(1)=1  
MSTP(81)=20

ME correction on/off no change, to check more.

# 2nd example: pythia8 built-in DY

```
----- PYTHIA Event Listing (hard process) -----
no      id   name      status   mothers   daughters   colours   p_x   p_y   p_z   e       m
0       90  (system)    -11      0       0       0       0       0.000  0.000  0.000  13000.000 13000.000
1      2212 (p+)      -12      0       0       3       0       0       0.000  0.000  6500.000 6500.000  0.938
2      2212 (p+)      -12      0       0       4       0       0       0.000  0.000  -6500.000 6500.000  0.938
3       2 (u)        -21      1       0       5       0       101     0       0.000  0.000  2383.952 2383.952  0.000
4      -2 (ubar)     -21      2       0       5       0       0       101     0.000  0.000  -0.784   0.784   0.000
5      23 (Z0)        -22      3       4       6       7       0       0       0.000  0.000  2383.167 2384.736  86.486
6       5 b          23      5       0       0       0       102     0       -37.262 -7.002  1749.515 1749.933  4.800
7      -5 bbar       23      5       0       0       0       0       102     37.262  7.002   633.652  634.803  4.800
Charge sum: 0.000
Momentum sum: 0.000
0.000
0.000
2383.167
2384.736
86.486
----- End PYTHIA Event Listing -----
```

```
----- PYTHIA Event Listing (complete event) -----
no      id   name      status   mothers   daughters   colours   p_x   p_y   p_z   e       m
0       90  (system)    -11      0       0       0       0       0.000  0.000  0.000  13000.000 13000.000
1      2212 (p+)      -12      0       0       314     0       0       0.000  0.000  6500.000 6500.000  0.938
2      2212 (p+)      -12      0       0       315     0       0       0.000  0.000  -6500.000 6500.000  0.938
3       2 (u)        -21      6       6       5       0       101     0       0.000  0.000  2383.952 2383.952  0.000
4      -2 (ubar)     -21      7       0       5       0       0       101     0.000  0.000  -0.784   0.784   0.000
5      23 (Z0)        -22      3       4       8       8       0       0       0.000  0.000  2383.167 2384.736  86.486
6       2 (u)        -42      10      10      3       3       101     0       -0.000 0.000  2383.952 2383.952  0.000
7      21 (g)         -41      11      0       9       4       103     101    0.000  -0.000  -3.999   3.999  0.000
8      23 (Z0)        -44      5       5       12      12      0       0       68.581  -21.319  1810.674 1814.161  86.486
9       2 (u)        -43      7       0       13      13      103     0       -68.581 21.319   569.278  573.790  0.330
10      2 (u)        -42      15      15      6       6       101     0       -0.000 0.000  2383.952 2383.952  0.000
11      21 (g)        -41      16      0       14      7       104     101    0.000  -0.000  -30.073  30.073  0.000
12      23 (Z0)        -44      8       8       17      17      0       0       55.699  -19.505  1528.996 1532.577  86.486
13       2 (u)        -44      9       9       18      18      103     0       -85.253 23.666  842.271  846.906  0.330
14      21 (g)        -43      11      0       19      19      104     103    29.555  -4.160   -17.389  34.542  0.000
15       2 (u)        -42      25      0       10      10      101     0       0.000  0.000  2383.952 2383.952  0.000
16       3 (s)        -41      26      26      20      11      104     0       -0.000 0.000  -694.563  694.563  0.000
17      23 (Z0)        -44      12      12      27      27      0       0       55.079  -20.248  1522.072 1525.656  86.486
18       2 (u)        -44      13      13      28      28      103     0       -86.055 22.705  851.453  856.092  0.330
-----
```

# 2nd example: pythia8 built-in DY

315	3	(s)	-61	2	0	256	256	150	0	-0.246	-2.225	-695.233	695.237	0.000	
316	23	(Z0)	→	-62	91	91	419	420	0	0	53.429	-20.200	1524.509	1528.029	86.486
317	2	(u)		-62	92	92	578	578	110	0	-24.537	9.992	258.001	259.358	0.330
318	21	(g)		-62	251	251	528	528	164	174	16.057	-10.177	-14.947	24.183	0.000

418	-2	(ubar)	-63	2	0	459	459	0	195	0.157	0.382	-408.655	408.655	0.330	
419	5	(b)		-23	316	0	421	422	102	0	1.927	-21.818	1119.720	1119.944	4.800
420	-5	(bbar)		-23	316	0	423	423	0	102	51.503	1.618	404.790	408.084	4.800

991	22	gamma	91	981	0	0	0	0	0	0.127	-0.289	11.057	11.062	0.000
992	22	gamma	91	981	0	0	0	0	0	0.079	-0.150	3.354	3.358	0.000
993	111	(pi0)	-91	982	0	1001	1002	0	0	-0.321	-0.911	78.347	78.353	0.135
994	211	pi+	91	982	0	0	0	0	0	0.179	-2.507	133.098	133.122	0.140
995	2114	(Delta0)	-91	982	0	1003	1004	0	0	0.340	-5.478	317.728	317.778	1.254
996	211	pi+	91	988	0	0	0	0	0	2.466	0.027	21.944	22.082	0.140
997	-211	pi-	91	988	0	0	0	0	0	3.718	0.297	31.910	32.128	0.140
998	111	(pi0)	-91	988	0	1005	1006	0	0	4.303	0.110	37.037	37.287	0.135
999	22	gamma	91	990	0	0	0	0	0	1.091	0.074	8.671	8.740	0.000
1000	22	gamma	91	990	0	0	0	0	0	3.897	0.276	33.098	33.328	0.000
1001	22	gamma	91	993	0	0	0	0	0	-0.225	-0.538	51.649	51.652	0.000
1002	22	gamma	91	993	0	0	0	0	0	-0.095	-0.373	26.697	26.700	0.000
1003	2212	p+	91	995	0	0	0	0	0	0.258	-3.609	199.953	199.988	0.938
1004	-211	pi-	91	995	0	0	0	0	0	0.082	-1.869	117.775	117.790	0.140
1005	22	gamma	91	998	0	0	0	0	0	3.821	0.103	33.242	33.461	0.000
1006	22	gamma	91	998	0	0	0	0	0	0.482	0.007	3.795	3.825	0.000
Charge sum: 2.0000 Momentum sum: 0.0000 -0.0000 0.0000 13000.0000 13000.0000														
----- End PYTHIA Event Listing -----														

## 2nd example: pythia8 built-in DY

Status Code:

Multi-Copies

Z: -22, -22, -44, ..., -44, -62

<http://home.thep.lu.se/~torbjorn/pythia81html/ParticleProperties.html>

21 - 29 : particles of the hardest subprocess

21 : incoming

22 : intermediate (intended to have preserved mass)

23 : outgoing

24 : outgoing, nonperturbatively kicked out in diffraction

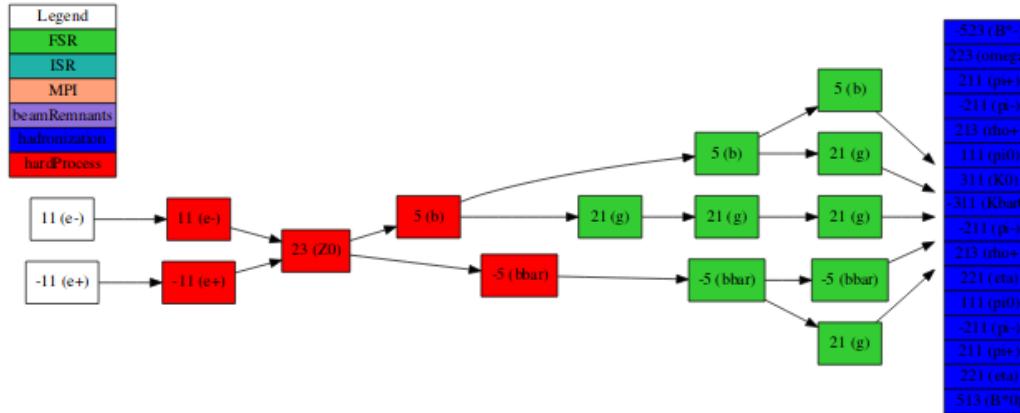
41 - 49 : particles produced by initial-state-showers

61 - 69 : particles produced by beam-remnant treatment

# Parton Shower Event Display

New users often wonder what goes on in PYTHIA. We included a bit of silly visualization for that, see `include/Pythia8Plugins/Visualization.h`

LEP looks pretty simple...

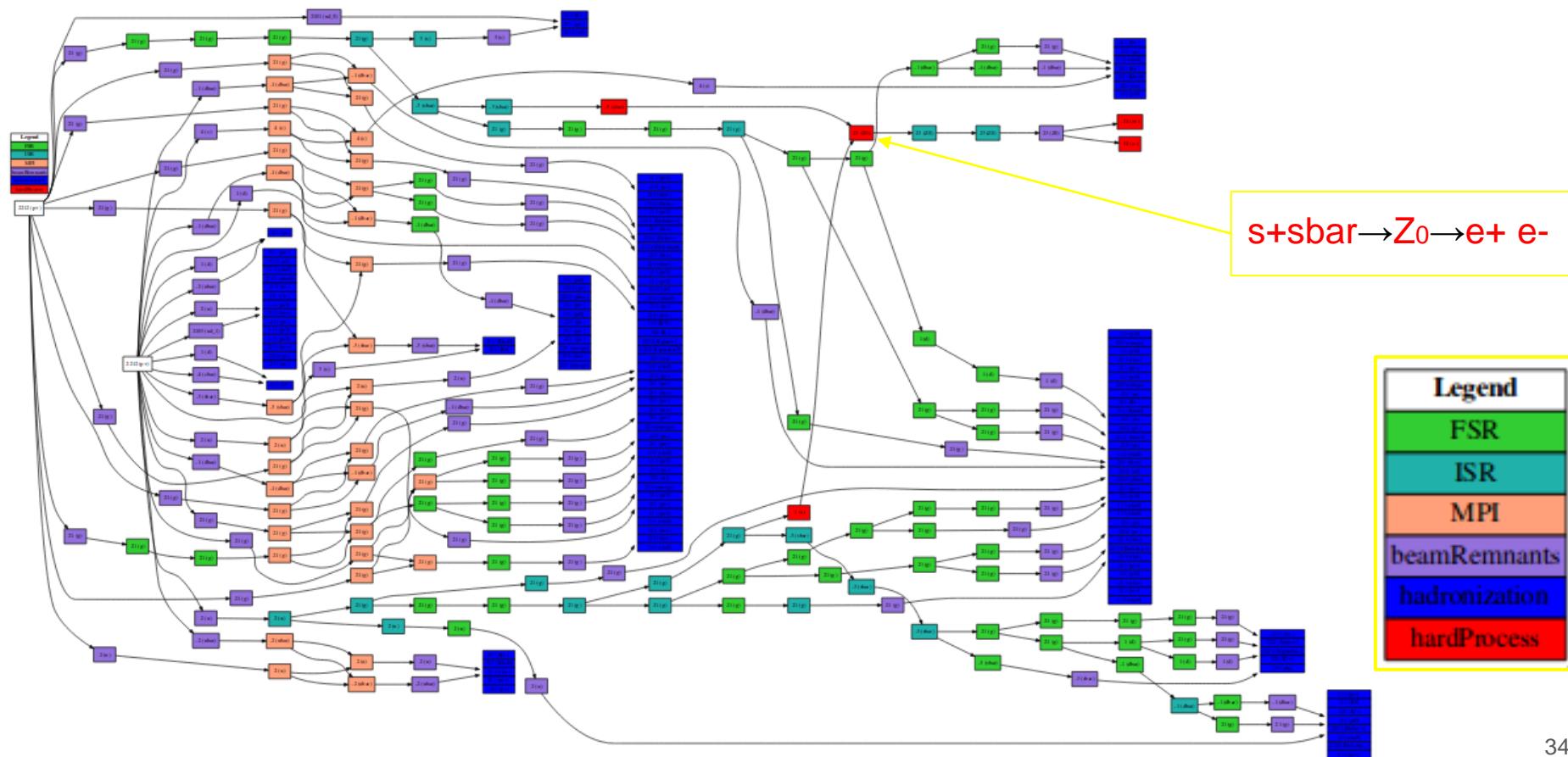


<http://home.thep.lu.se/~torbjorn/pythia83html/ExampleKeywords.html>

Generated with...

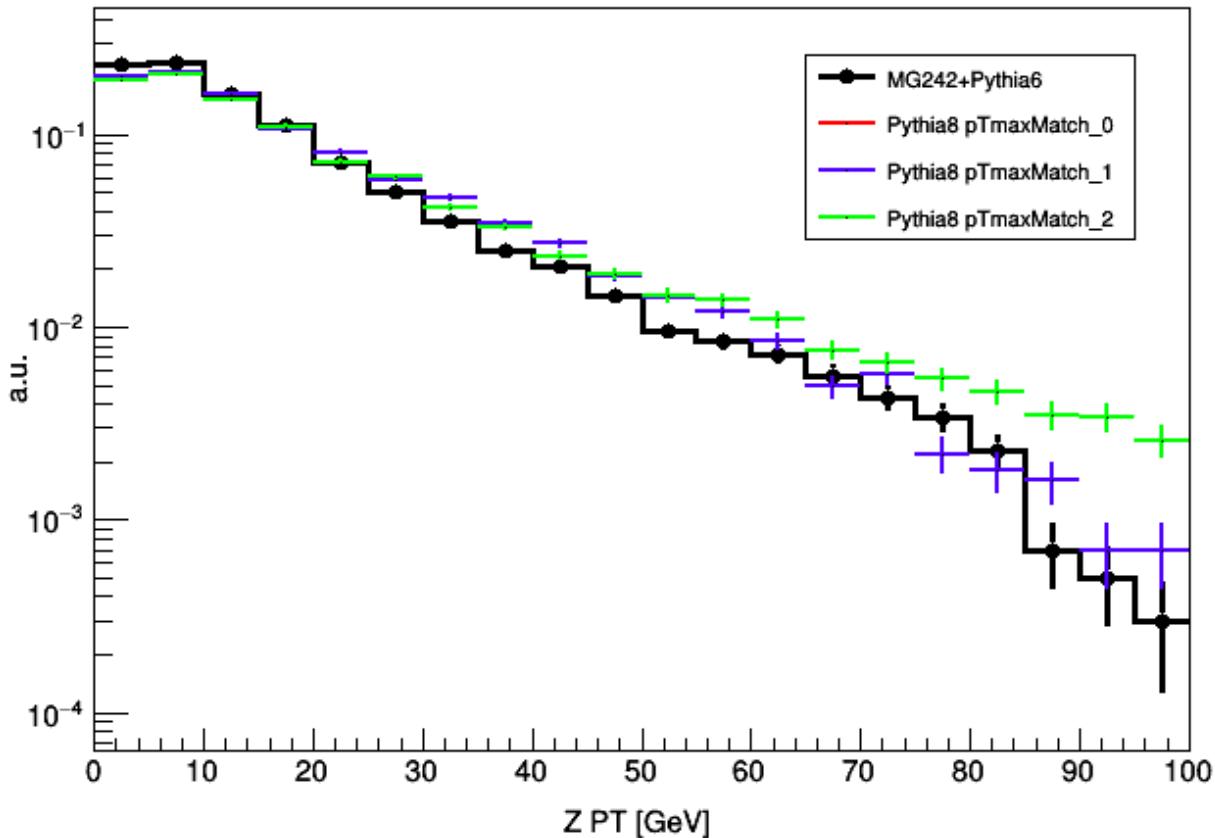
```
./main300 --input main300.cmnd --visualize_event
```

# Parton Shower Event Display



# 2nd example: pythia8 built-in DY

$pp \rightarrow Z \rightarrow \mu^+\mu^-$  at 13TeV LHC



wimpy  
vs  
power  
shower

TimeShower:pTmaxMatch = 0/1/2  
SpaceShower:pTmaxMatch = 0/1/2

## 2nd example: pythia8 built-in DY

PDF/Tune

<http://home.thep.lu.se/~torbjorn/pythia82html/PDFSelection.html>

PDF:pSet (default = 13)

option 13 : NNPDF2.3 QCD+QED LO alpha\_s(M\_Z) = 0.130.

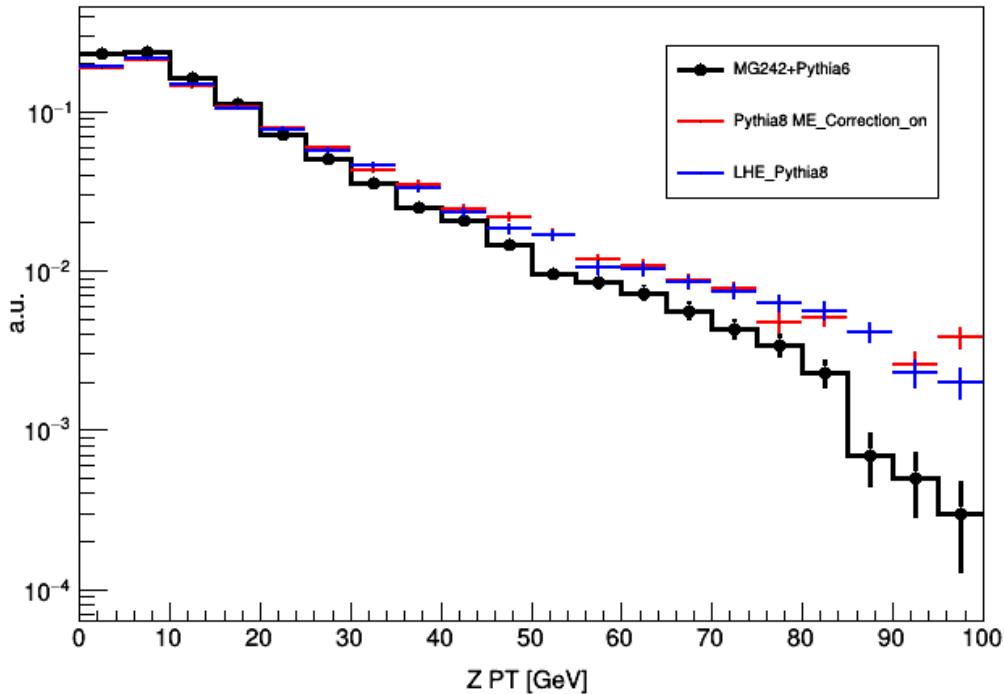
<http://home.thep.lu.se/~torbjorn/pythia81html/Tunes.html>

mode Tune:pp (default = 5; minimum = -1; maximum = 14)

Since some physics aspects cannot be derived from first principles, this program contains many parameters that represent a true uncertainty in our understanding of nature. Particularly afflicted are the areas of hadronization and multiparton interactions, which both involve nonperturbative QCD physics.

## 2nd example: lhe → Pythia8

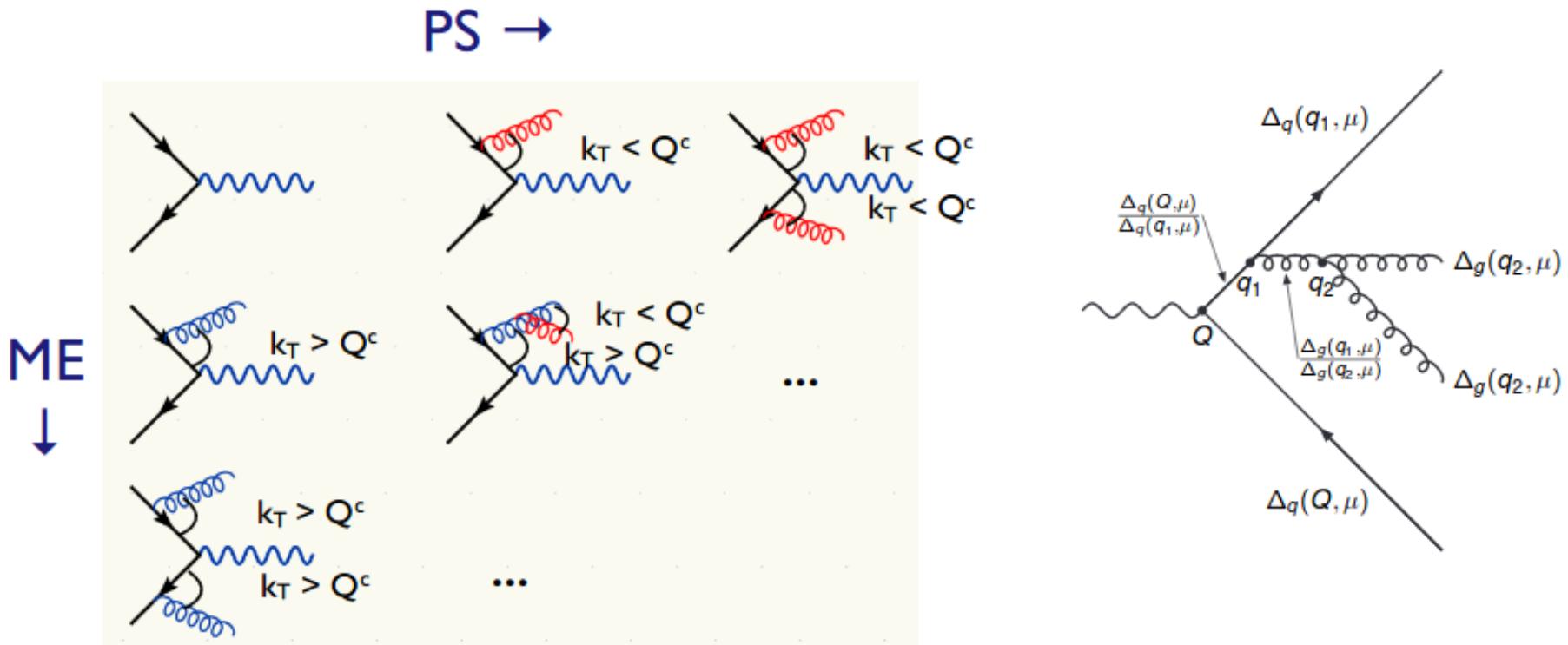
$pp \rightarrow z \rightarrow \mu^+\mu^-$  at 13TeV LHC



main11.cc

```
pythia.readString("Beams:frameType = 4");
if (useGzip) pythia.readString("Beams:LHEF =
ttbar.lhe.gz");
else         pythia.readString("Beams:LHEF =
ttbar.lhe");
pythia.init();
```

# 3rd example: MLM matching



<http://hep.ucsb.edu/people/cag/Matching.pdf>

[https://www.physik.uzh.ch/~marek/talks/20170910\\_Corfu.pdf](https://www.physik.uzh.ch/~marek/talks/20170910_Corfu.pdf)

# 3rd example: MLM matching

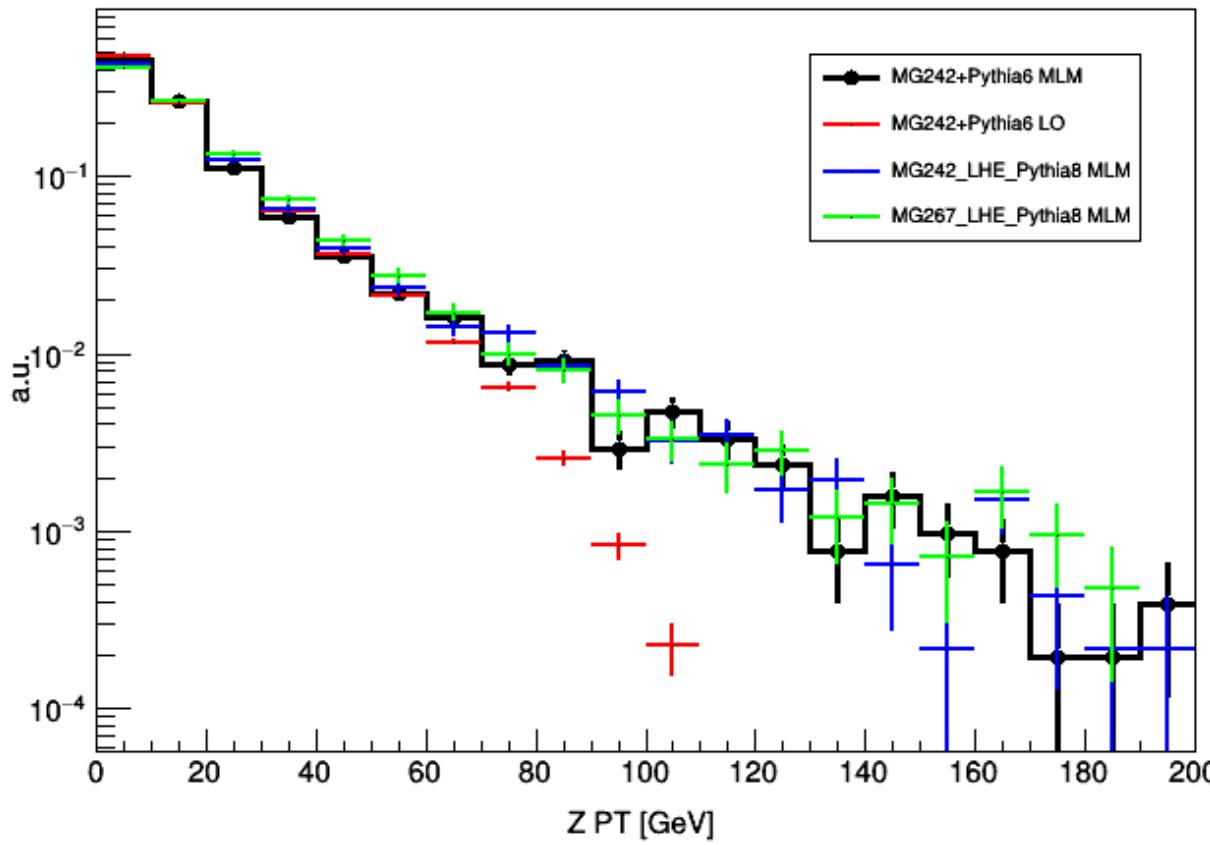
[https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test1\\_main32-MLM/main32.cc](https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test1_main32-MLM/main32.cc)

```
// Generator and read in commands.  
Pythia pythia;  
pythia.readFile("main32.cmdn");  
  
// Extract settings to be used in the main program.  
int nEvent = pythia.mode("Main:numberOfEvents");  
int nAbort = pythia.mode("Main:timesAllowErrors");  
int nSkip  = pythia.mode("Main:spareMode1");  
  
// Create UserHooks pointer. Stop if it failed. Pass pointer to Pythia.  
CombineMatchingInput combined;  
UserHooks* matching = combined.getHook(pythia);  
if (!matching) return 1;  
pythia.setUserHooksPtr(matching);
```

! 3) Enable matching  
JetMatching:merge = on  
  
! Madgraph run:  
Beams:LHEF = z012mlm.lhe  
Beams:frameType = 4  
JetMatching:scheme = 1  
JetMatching:setMad = off  
JetMatching:qCut = 20.0  
JetMatching:coneRadius = 1.0  
JetMatching:etaJetMax = 10.0  
JetMatching:nJetMax = 2

# 3rd example: MLM matching

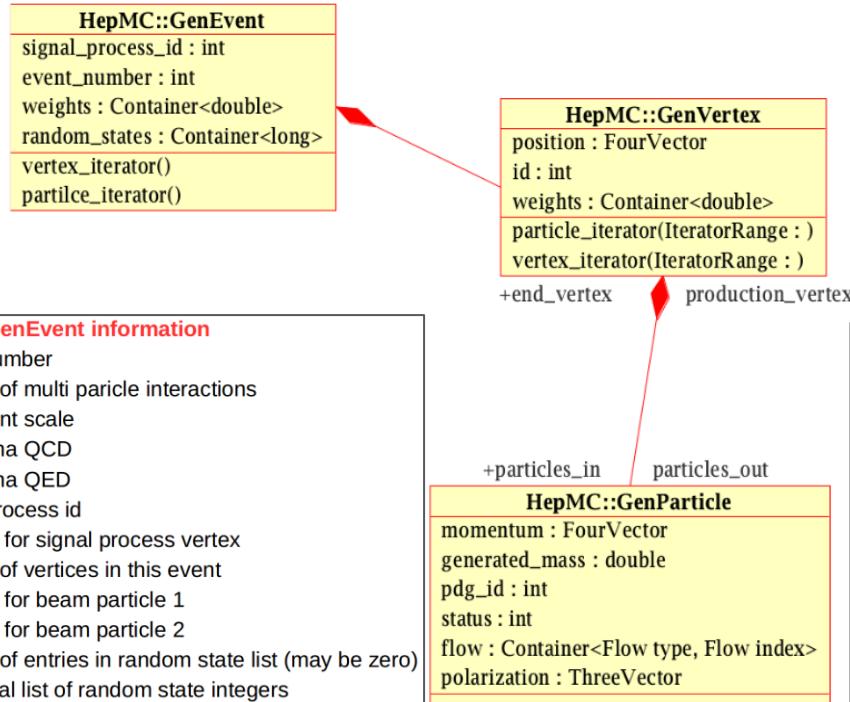
$pp \rightarrow z \rightarrow \mu^+\mu^-$  012Jets MLM at 13TeV LHC



# 4th example: hepmc analysis

MadGraph+Pythia will produce the hepmc file, how to analyze it?

<http://lcgapp.cern.ch/project/simu/HepMC/205/html/examples.html>



## E - general GenEvent information

- **int:** event number
- **int:** number of multi particle interactions
- **double:** event scale
- **double:** alpha QCD
- **double:** alpha QED
- **int:** signal process id
- **int:** barcode for signal process vertex
- **int:** number of vertices in this event
- **int:** barcode for beam particle 1
- **int:** barcode for beam particle 2
- **int:** number of entries in random state list (may be zero)
- **long:** optional list of random state integers
- **int:** number of entries in weight list (may be zero)
- **double:** optional list of weights

## V - GenVertex information

- **int:** barcode
- **int:** id
- **double:** x
- **double:** y
- **double:** z
- **double:** ctau
- **int:** number of "orphan" incoming particles
- **int:** number of outgoing particles
- **int:** number of entries in weight list (may be zero)
- **double:** optional list of weights

## P - GenParticle information

- **int:** barcode
- **int:** PDG id
- **double:** px
- **double:** py
- **double:** pz
- **double:** energy
- **double:** generated mass
- **int:** status code
- **double:** Polarization theta
- **double:** Polarization phi
- **int:** barcode for vertex that has this particle as an incoming particle
- **int:** number of entries in flow list (may be zero)
- **int, int:** optional code index and code for each entry in the flow list

# 4th example: hepmc analysis

Pythia8 can also output hepmc file

```
// Interface for conversion from Pythia8::Event to HepMC event.  
HepMC::Pythia8ToHepMC ToHepMC;  
  
// Specify file where HepMC events will be stored.  
HepMC::IO_GenEvent ascii_io(argv[2], std::ios::out);  
  
// Construct new empty HepMC event and fill it.  
// Units will be as chosen for HepMC build, but can be changed  
// by arguments, e.g. GenEvt( HepMC::Units::GEV, HepMC::Units::MM)  
HepMC::GenEvent* hepmcevt = new HepMC::GenEvent();  
ToHepMC.fill_next_event( pythia, hepmcevt );  
  
// Write the HepMC event to file. Done with it.  
ascii_io << hepmcevt;  
delete hepmcevt;
```

# 4th example: hepmc analysis

迭代器：

## 1.GenEvent

particle 迭代器：能逐个取到每个 event 中的每个 particle

```
HepMC::GenEvent::particle_const_iterator p = evt->particles_begin()
```

vertex 迭代器：能逐个取到每个 event 中的每个 vertex

```
HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();v != evt->vertices_end()
```

## 2.GenVertex

vertex 中的 particle 迭代器：能逐个取到每个 vertex 中的每个 particle

// 出射粒子

```
HepMC::GenVertex::particles_out_const_iterator p = (*v)->particles_out_const_begin()
```

// 入射粒子

```
HepMC::GenVertex::particles_in_const_iterator p = (*v)->particles_in_const_begin()
```

# 4th example: hepmc analysis

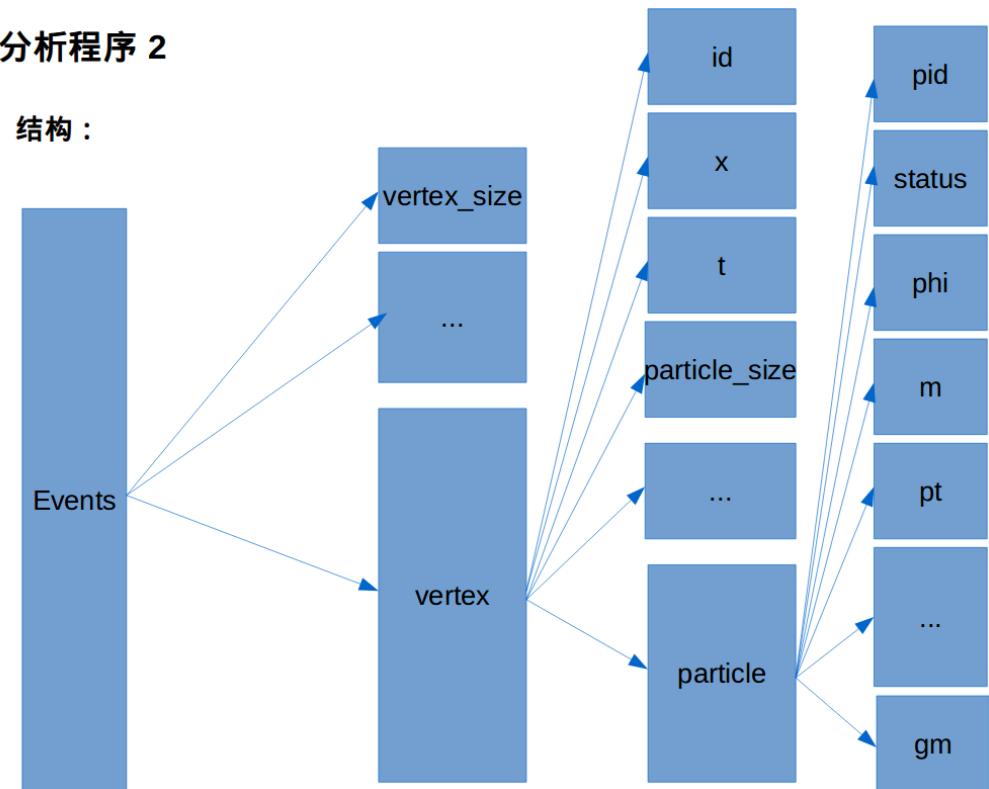
分析程序 1

```
//vertex
id = (*v)->id();
x = (*v)->position().x();
y = (*v)->position().y();
z = (*v)->position().z();
t = (*v)->position().t();

//particle
status = (*p)->status();
pid = (*p)->pdg_id();
pt = (*p)->momentum().perp();
px = (*p)->momentum().px();
py = (*p)->momentum().py();
pz = (*p)->momentum().pz();
m = (*p)->momentum().m();
gm = (*p)->generated_mass();
theta = (*p)->polarization().theta();
phi = (*p)->polarization().phi();
```

分析程序 2

结构：



# 4th example: hepmc analysis

[https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test2\\_hepmcanalysis/testmine1.cc](https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test2_hepmcanalysis/testmine1.cc)

```
for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin(); v != evt->vertices_end(); ++v ) {
    if(isw!=0) cout<<"--"<<endl;
    for ( HepMC::GenVertex::particles_out_const_iterator p = (*v)->particles_out_const_begin(); p!=(*v)
        isw=0;
        if((*p)->pdg_id()==23){
            isw=1;
            vbarcode          =      (*p)->end_vertex()->barcode();
            cout<<" "<<vbarcode<<" "<<(*p)->status()<<endl;
            if((*p)->status()==62)  {
                ptz      =      (*p)->momentum().perp();
                ez       =      (*p)->momentum().e();
                h->Fill();
            }
        }
    }
}
```

Loop over vertices and gen particles to find the last Z0

# 4th example: hepmc analysis

[https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test2\\_hepmcanalysis/testmine2.cc](https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test2_hepmcanalysis/testmine2.cc)

```
while ( evt ) {  
    EveN++;  
    for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin(); p!=evt->particles_end(); ++p )  
        if((*p)->pdg_id()==23){  
            vbarcode      =      (*p)->end_vertex()->barcode();  
            cout<<" "<<vbarcode<<" "<<(*p)->status()<<endl;  
            if((*p)->status()==62) {  
                ptz      =      (*p)->momentum().perp();  
                ez       =      (*p)->momentum().e();  
                h->Fill();  
            }  
        }  
}
```

Loop over all gen particles and use status code  
to find the last Z0

# 4th example: hepmc analysis

[https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test2\\_hepmcanalysis/testmine3.cc](https://github.com/qliphi/2020HEPMC/blob/master/pythia8/test2_hepmcanalysis/testmine3.cc)

```
const HepMC::GenParticle* getlastcopy(const HepMC::GenParticle* mother)
{
    if ( !(mother->end_vertex()) && (mother->status() != 2) ) { //final state
        return mother;
    }
    else {
        if ( !(mother->end_vertex()) ) { //something is wrong in HepMC
            return mother;
        };
    };
    HepMC::GenVertex::particle_iterator firstChild, thisChild, lastChild;
    firstChild = mother->end_vertex()->particles_begin(HepMC::children);
    lastChild = mother->end_vertex()->particles_end(HepMC::children);
    if ((*firstChild)->pdg_id() == (mother)->pdg_id()) {
        return getlastcopy(*firstChild);
    }
    else {
        return mother;
    }
}

for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin(); p!=evt->particles_end(); ++p ){
    if((*p)->pdg_id()==23){

        const HepMC::GenParticle* mother1 = getfirstcopy(*p);
        cout<<"mother1 "<<mother1->pdg_id()<< " "<<mother1->status()<<endl;
        std::vector<const HepMC::GenParticle*> mothers = getMothers(mother1);
        cout<<"mothers "<<mothers.size()<< " "<<mothers[0]->pdg_id()<< " "<<mothers[1]->pdg_id()<<endl;

        const HepMC::GenParticle* daughter1 = getlastcopy(*p);
        cout<<"daughter1 "<<daughter1->pdg_id()<< " "<<daughter1->status()<<endl;
        std::vector<const HepMC::GenParticle*> daughters = getDaughters(daughter1);
        cout<<"daughters "<<daughters.size()<< " "<<daughters[0]->pdg_id()<< " "<<daughters[1]->pdg_id()<<endl;

        ptz      =      daughter1->momentum().perp();
        ez       =      daughter1->momentum().e();
        h->Fill();
        continue;
    }
}
```

Define the first and last copy

# 4th example: hepmc analysis

[https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test2\\_hepmcanalysis/testmine3.cc](https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test2_hepmcanalysis/testmine3.cc)

Define the first and last copy

this Z 22

mother1 23 22

mothers 2 -2 2

daughter1 23 62

daughters 2 -13 13

this Z 44

mother1 23 22

mothers 2 -2 2

daughter1 23 62

daughters 2 -13 13

this Z 44

mother1 23 22

mothers 2 -2 2

daughter1 23 62

daughters 2 -13 13

this Z 62

mother1 23 22

mothers 2 -2 2

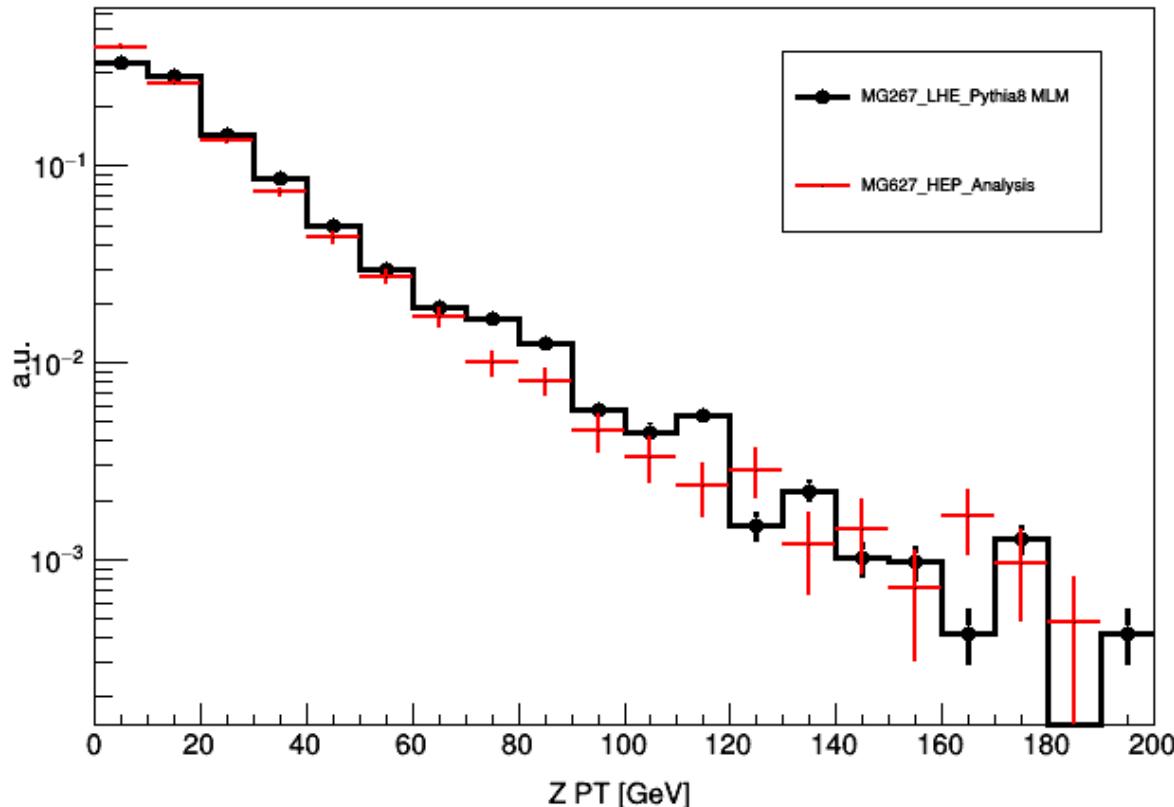
daughter1 23 62

daughters 2 -13 13

For a random chosen event,  
Print out for each Z  
Its current status,  
Its first mother's information,  
and  
Its last copy's information.

# 4th example: hepmc analysis

$pp \rightarrow Z \rightarrow \mu^+\mu^-$  012Jets MLM at 13TeV LHC



MG->LHE->Pythia8->Analysis

MG+Pythia8->Hepmc->Analysis

# 5th example: pythia8 status

[https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test3\\_p8status/main02.cc](https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test3_p8status/main02.cc)

```
// Begin event loop. Generate event. Skip if error. List first one.  
for (int iEvent = 0; iEvent < 10000; ++iEvent) {  
    if (!pythia.next()) continue;  
    // Loop over particles in event. Find last Z0 copy. Fill its pT.  
    int iZ = 0;  
    cout<<"-----"<<endl;  
    for (int i = 0; i < pythia.event.size(); ++i)  
        if (pythia.event[i].id() == 23) {  
            iZ = i;  
  
            cout<<"m1 "<<pythia.event[iZ].mother1()<<" "<<pythia.event[pythia.event[iZ].mother1()].id()<<" "<<pythia.e  
            cout<<"d1 "<<pythia.event[iZ].daughter1()<<" "<<pythia.event[pythia.event[iZ].daughter1()].id()<<" "<<pythi  
            status=pythia.event[i].status();  
            t2->Fill();  
        }  
  
    cout<<"my "<<getfirstcopy(pythia.event, iZ)<<" "<<getlastcopy(pythia.event, iZ)<<endl;  
  
    ptz=pythia.event[iZ].pT();  
    ez=pythia.event[iZ].e();  
    // End of event loop. Statistics. Histogram. Done.  
    t->Fill();  
}
```

```
int getfirstcopy( Event& event, int i ) {  
    if (event[event[i].mother1()].id() == event[i].id()) {  
        return getfirstcopy(event, event[i].mother1());  
    }  
    else {  
        return i;//event[i].mother1();  
    }  
}  
  
int getlastcopy( Event& event, int i ) {  
    if (event[event[i].daughter1()].id() == event[i].id()) {  
        return getlastcopy(event, event[i].daughter1());  
    }  
    else {  
        return i;//event[i].daughter1();  
    }  
}  
-----<<endl;  
]) . id() << endl;
```

# 6th example: Higgs decay

[https://github.com/qliphy/2020HEPMC/tree/master/pythia8/test4\\_Hdecay](https://github.com/qliphy/2020HEPMC/tree/master/pythia8/test4_Hdecay)

Starting from MadGraph

import model heft

generate p p > h      -> pythia8 to get hepmc file -> counting over Higgs decay

```
for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin(); p!=evt->particles_end(); ++p ){
    if((*p)->pdg_id()==25){
        const HepMC::GenParticle* daughter1 = getlastcopy(*p);
        std::vector<const HepMC::GenParticle*> daughters = getDaughters(daughter1);
        //cout<<"daughters "<<daughters.size()<< " "<<daughters[0]->pdg_id()<< " "<<daughters[1]->pdg_id()
        if(abs(daughters[0]->pdg_id())==5 & abs(daughters[1]->pdg_id())==5) {
            hbb++;
        }
        else if(abs(daughters[0]->pdg_id())==24 & abs(daughters[1]->pdg_id())==24) {
            hww++;
        }
        else if(abs(daughters[0]->pdg_id())==23 & abs(daughters[1]->pdg_id())==23) {
            hzz++;
        }
        else if(abs(daughters[0]->pdg_id())==21 & abs(daughters[1]->pdg_id())==21) {
            hgg++;
        }
    }
}
```

# 6th example: Higgs decay

```
hbb=568 hww=218 hzz=29 hgg=87 htautau=66 haa=0 hcc=29 hmm=1 hza=2
```

```
ntot=1000 1000
```

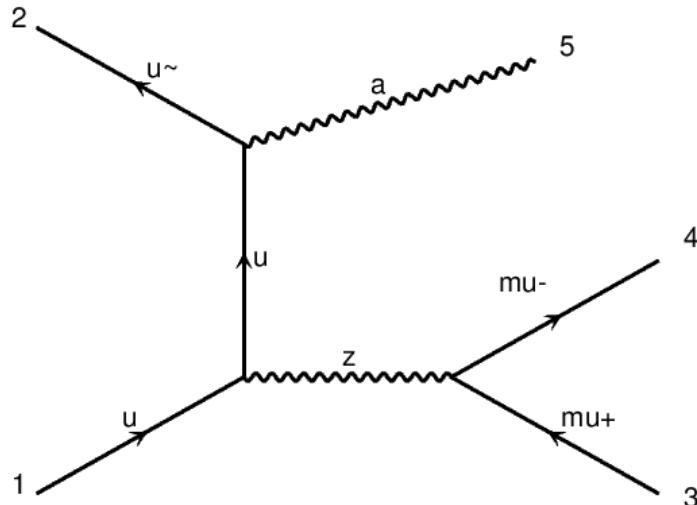
```
hbb=0.568 hww=0.218 hzz=0.029 hgg=0.087 htautau=0.066 haa=0 hcc=0.029 hmm=0.029 hza=0.002
```

<http://home.thep.lu.se/~torbjorn/pythia82html/ParticleDataScheme.html>

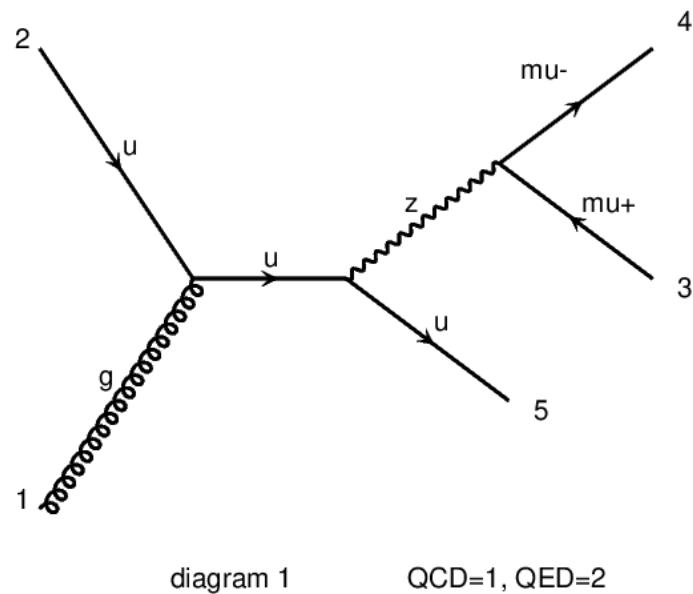
Decay channel can be specified, e.g.

```
'SLHA:useDecayTable = off',
'25:m0 = 125.0',
'25:onMode = off',
'25:onIfMatch = 24 -24',      # turn ON H->WW
'24:mMin = 0.05',            #
'24:onMode = off',           # turn OFF all W decays
'24:onNeglIfAny = 11 13 15 12 14 16', # turn ON W- -> lnu
'24:onPoslIfAny = 1 2 3 4 5'          # turn ON W+ -> qq
```

# 7th example: Zgamma vs ZJets



Photon directly produced from matrix-element



No photon from matrix-element,  
However, can be produced from  
Parton shower: ISR, FSR

# 7th example: Zgamma vs ZJets

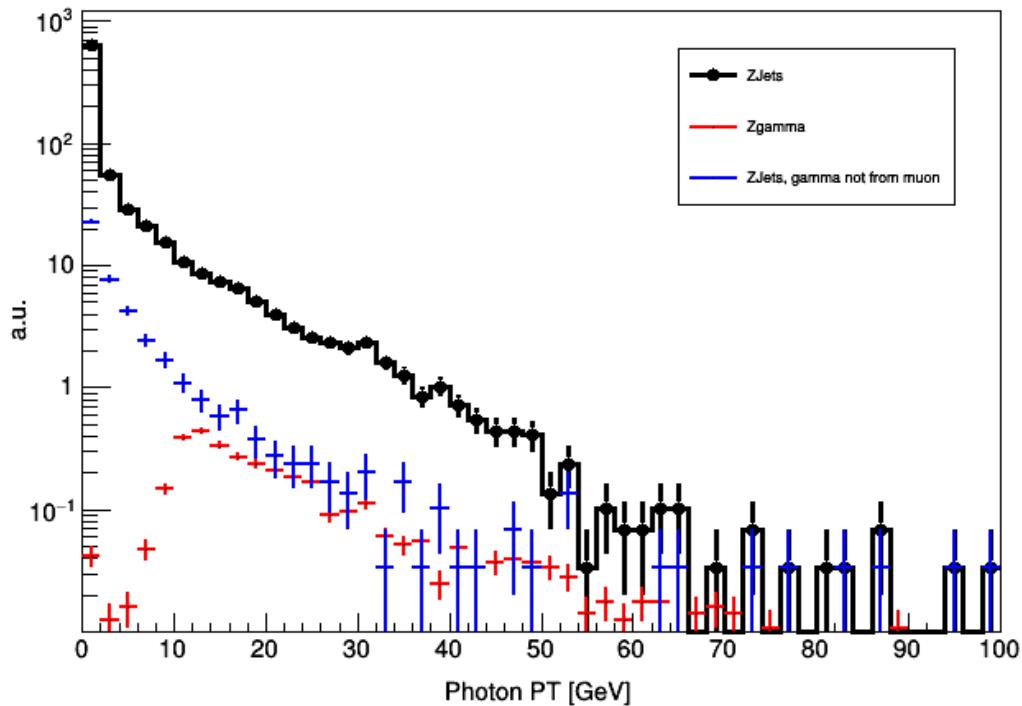
[https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test\\_photon/testmine3.cc](https://github.com/qliphy/2020HEPMC/blob/master/pythia8/test_photon/testmine3.cc)

```
if((*p)->pdg_id()==22 && (*p)->status()==1 ){
    const HepMC::GenParticle* mother1 = getfirstcopy(*p);
    //cout<<"mother1 "<<mother1->pdg_id()<<" "<<mother1->status()<<endl;
    std::vector<const HepMC::GenParticle*> mothers = getMothers(mother1);
    cout<<"mothers "<<mothers.size()<<" "<<mothers[0]->pdg_id()<<endl;
    int asta=abs(mothers[0]->pdg_id());
    if((asta>0 && asta<7) || (asta>10 && asta<17) || asta==24 ) {
        if((asta>0 && asta<7) || (asta>10 && asta<17 && asta!=13) || asta==24 ) {
            if (pta < (*p)->momentum().perp()) {
                pta = (*p)->momentum().perp();
            }
        }
    }
}
```

Only selecting photon radiating from quarks, leptons, and W bosons,  
Not those from pi0 decay

# 7th example: Zgamma vs ZJets

ZJets vs Z+Photon at 13TeV LHC



## Note for “Zgamma”

- generate  $p\ p > z\ a$ ,  $z > \mu+\mu-$   
Photons from FSR, i.e. muon  
radiation are not included.
- Moreover in run\_card.dat,  
10.0 = pta

# 8th example: Rivet

Rivet — the particle-physics MC analysis toolkit.

One can also interface Hepmc with Rivet to make comparisons, with Data

ATLAS\_2015\_I1408516\_MU

$Z_{PT}$  and  $Z\phi^*$  in muon channel

Experiment: ATLAS (LHC)

Inspire ID: 1408516

Status: VALIDATED

Authors:

- Christian Gutschow

References:

- arXiv: 1512.02192
- submitted to EPJC

Beams: p+ p+

Beam energies: (4000.0, 4000.0) GeV

Run details:

- inclusive Z production in the muon channel

Distributions of transverse momentum  $p_T^{\ell\ell}$  and the angular variable  $\phi_\eta^*$  of Drell-Yan lepton pairs are measured in  $20.3 \text{ fb}^{-1}$  of proton-proton collisions at  $\sqrt{s} = 8 \text{ TeV}$  with the ATLAS detector at the LHC. Measurements in electron-pair and muon-pair final states are corrected for detector effects. Compared to previous measurements in proto-proton collisions at  $\sqrt{s} = 7 \text{ TeV}$ , these new measurements benefit from a larger data sample and improved control of systematic uncertainties. Measurements are performed in bins of lepton-pair mass above, around and below the  $Z$ -boson mass peak. Specify the lepton channel (default is Z->ee) by using the dedicated plugins ATLAS\_2015\_I1408516\_EL and ATLAS\_2015\_I1408516\_MU.

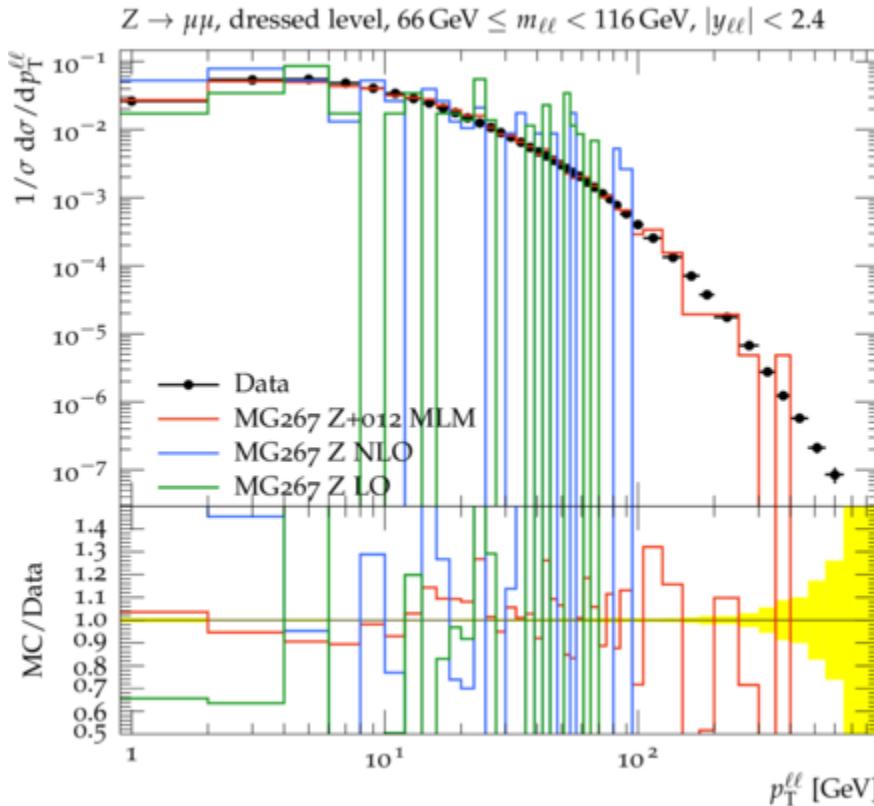
[https://rivet.hepforge.org/analyses/ATLAS\\_2015\\_I1408516\\_MU](https://rivet.hepforge.org/analyses/ATLAS_2015_I1408516_MU)



# 8th example: Rivet

rivet -a ATLAS\_2015\_I1408516\_MU mg267z012mlm.hepmc

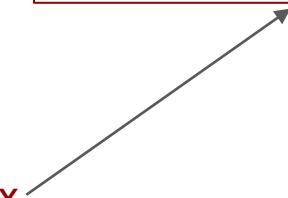
rivet-mkhtml Rivet1.yoda:'MG267 Z+012 MLM' Rivet2.yoda:'MG267 Z NLO'



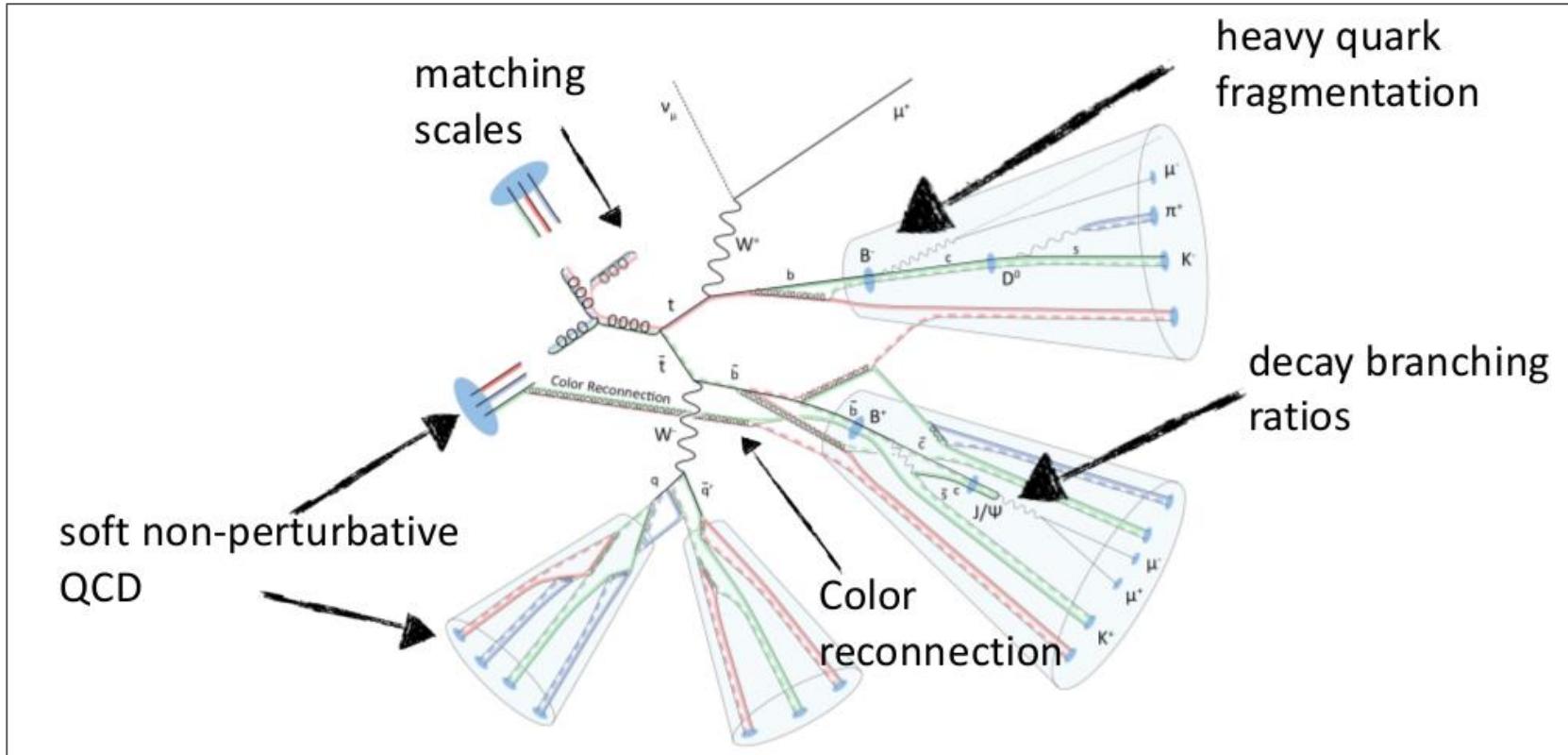
# Summary

- QCD and Parton Shower
- Pythia, Herwig, Sherpa
- Pythia6 example
- Pythia8
  - Event display
  - Drell Yan
  - MadGraph -> Pythia8: LO
  - MadGraph -> Pythia8: Matching
  - **MadGraph -> Pythia8: NLO, FxFx**
  - Hepmc
  - Status Code: H decay
  - Status Code: ZGamma vs ZJets
- Rivet

```
// main89.cc is a part of the PYTHIA event generator.  
// This program is written by Stefan Prestel.  
// It illustrates how to do run PYTHIA with LHEF input, allowing a  
// sample-by-sample generation of  
// a) Non-matched/non-merged events  
// b) MLM jet-matched events (kT-MLM, shower-kT, FxFx)  
// c) CKKW-L and UMEPS-merged events  
// d) UNLOPS NLO merged events  
// see the respective sections in the online manual for details.  
// ./main89 main89fxfx.cmnd 2.hep
```



# Go beyond



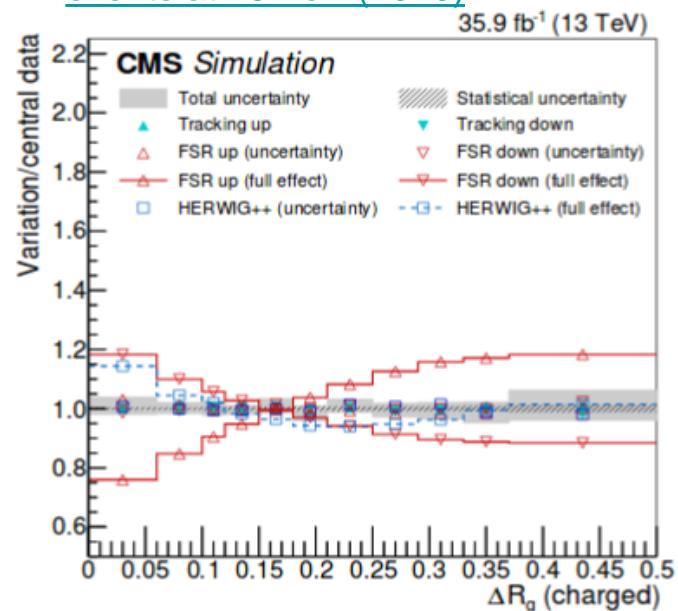
# Systematics

- Parton Shower Weights, Tune Variations etc included

$(\text{ISR}, \text{FSR}) \otimes (\mu_R, \text{cNS}) \otimes (g \rightarrow gg, g \rightarrow q\bar{q}, q \rightarrow qg, b/t \rightarrow b/t + g) \otimes (\text{up, down})$

Source	Handle	Weights	Variation	Note/Reference	Dedicated studies
Shower scales	ISR scale (SpaceShower:renormMultFac)	YES	0.5-2.0	FSR variations can be scaled down by $\sqrt{2}$ from LEP	TOP-15-011, TOP-16-021 TOP-17-13, TOP-17-015, ...
	FSR scale (TimeShower:renormMultFac)		0.5-2.0		
ME-PS Matching	hdamp	No	hdamp=1.58m <sub>t</sub> +0.66-0.59 m <sub>t</sub>	see TOP-16-021	Starting scale variations for MG5_aMC@NLO
	UE parameters	YES	UE tune up/down	See TOP-16-021 MPI & CR strength doesn't affect resonance decays	TOP-17-015 GEN-17-001
Color reconnection (odd clusters)	MPI based, QCD-inspired, gluon move	No	different models	CR affecting resonance decays	TOP-17-13, TOP-17-015
Fragmentation	momentum transfer from the b-quark to the B hadron: $x_b = p_T(B)/p_T(\text{b-jet})$	YES	Vary Bower-Lund parameter within uncertainties from LEP/SLD fits	see TOP-16-022 (re-weight $x_b$ )	
Flavor response/hadronization	Pythia vs Herwig	No	Vary the JES independently per flavour for light, g, c, b.		
Decay tables	B semi-leptonic BR	YES	vary semileptonic BR +0.77%/-0.45%	re-weight the fraction of semi-leptonic b jets by the PDG values (scale $\lambda_b$ to match PDG)	

CMS: [Jet substructure in ttbar events at 13 TeV \(2016\)](#)

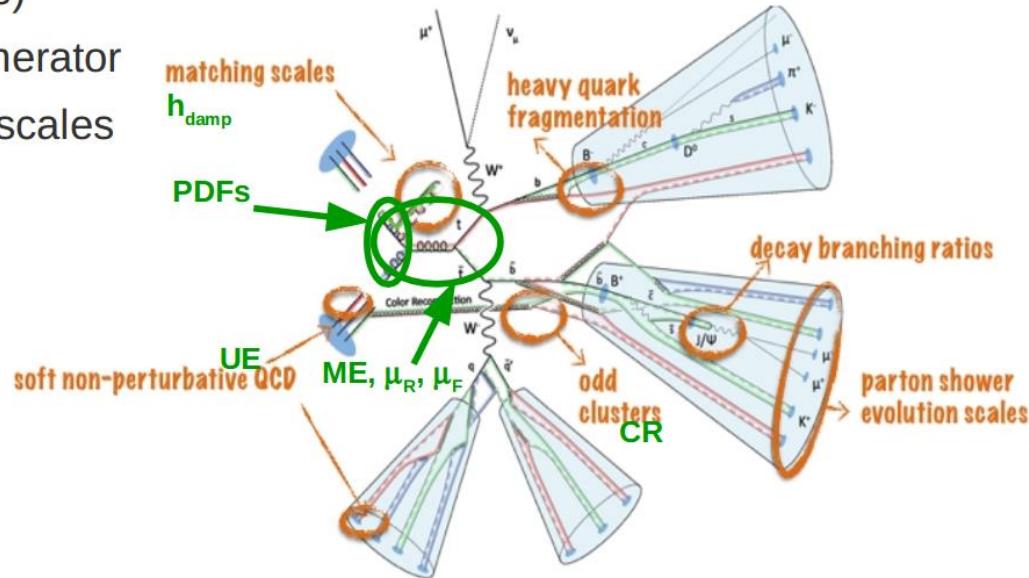


angle between the groomed subjets

# Top Quark Physics

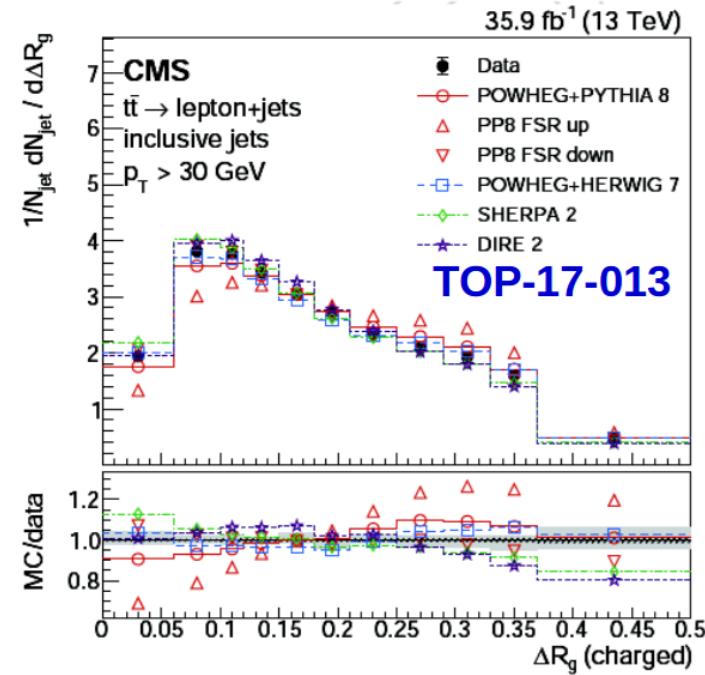
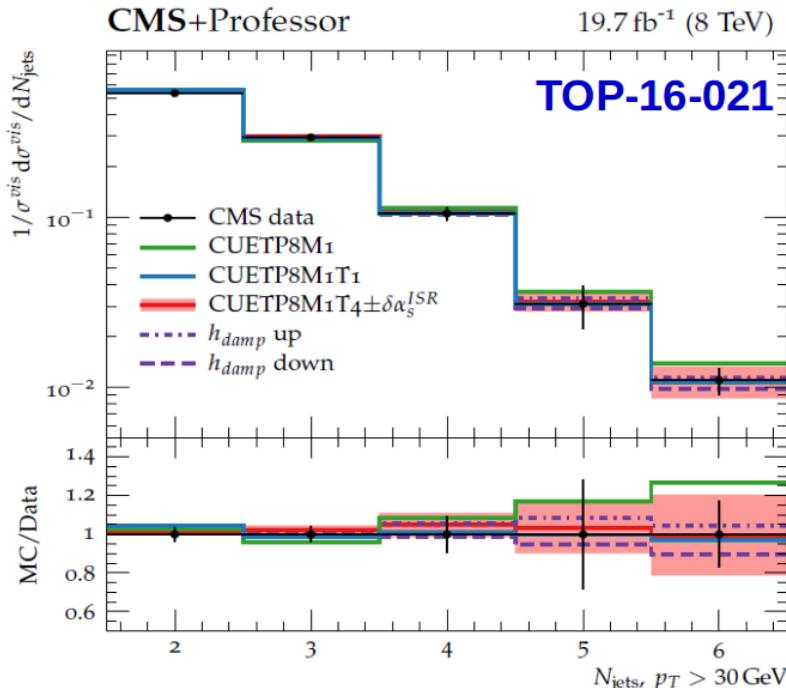
## Modeling uncertainties in top quark physics

- Parton distribution functions (PDFs)
- Choice of matrix element (ME) generator
- Factorization and renormalization scales
- ME-PS matching (aka ' $h_{\text{damp}}$ '')
- Parton shower (PS) scales
- Underlying event (UE)
- Color reconnection (CR)
- B quark fragmentation
- B hadron decays into leptons
- Top  $p_T$  reweighting
- Top mass
- Jet flavor response / hadronization



# ISR and FSR Scales

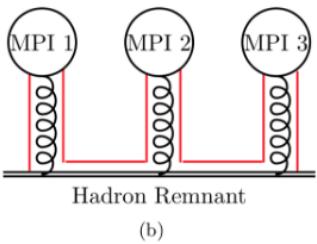
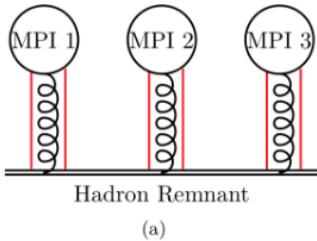
- Uncertainty on the value of  $\alpha_s(Q^2)$  in the parton shower  
 → initial and final state radiation (ISR and FSR)



**ISR:** recoil of the hard system,  $H_T$ ,  
 high- $p_T$  jet multiplicity, ...

**FSR:** jet broadening, out-of-cone energy, jet  
 multiplicity, ...

# Color Reconnection



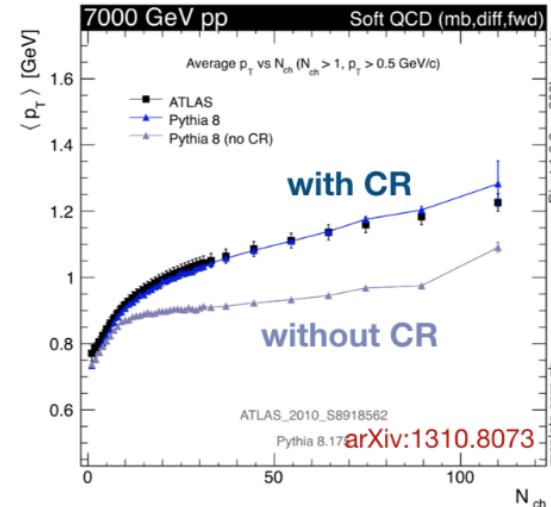
Leading colour approximation:

- each MPI is viewed as separate from all other systems in colour space.
- no strings stretched between different MPI systems.

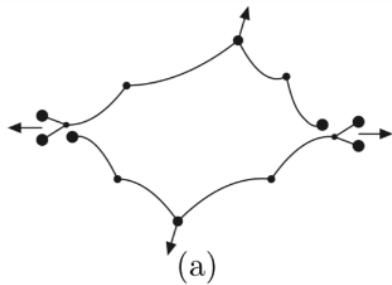
**Color reconnection** allows different MPI systems to be colour-connected to each other. MPI hadronize collectively.

Rising trend of  $\langle p_T \rangle(nch)$ :

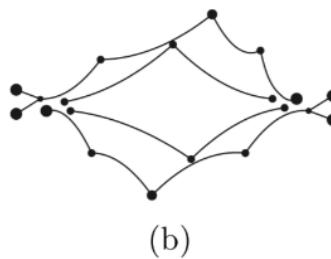
- first observed by UA1 (Nucl. Phys. B335 (1990))
- Color reconnection is needed to describe the data.



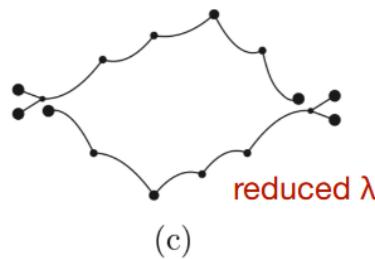
# MPI based Color-Reconnection



First hard scattering: outgoing gluons are colour connected



A second hard scattering: gives new strings connected to the remnants.



gluons are colour reconnected, so that the total string length ( $\lambda$ ) becomes as short as possible.

Reconnection probability

$$P_{\text{rec}}(p_T) = \frac{(R_{\text{rec}} \cdot p_{T0})^2}{R_{\text{rec}} \cdot p_{T0} + p_T^2}$$

ColourReconnection:range  
(free parameter)

The higher this number is the more reconnections can occur.

Reduce  $\lambda$  by adding partons of the lower- $p_T$  system to the strings defined by the higher- $p_T$  system

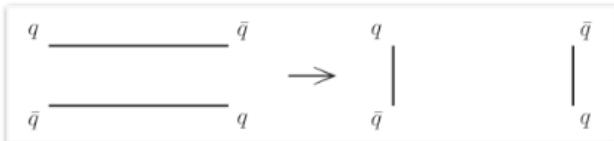
$$\lambda \approx \sum_{i,j} \ln \left( \frac{m_{ij}^2}{m_0^2} \right)$$

$p_T \downarrow \Rightarrow P_{\text{rec}} \uparrow$   
softer systems easier to reconnect

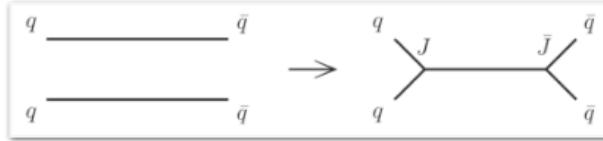
# QCD based Color Reconnection

Another approach for reconnection  $\Rightarrow$  introduce "junctions"

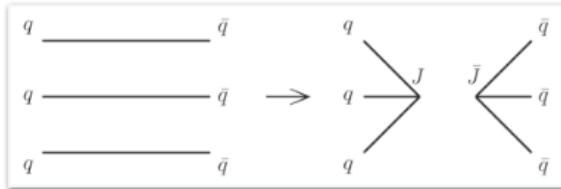
- More realistic; colour topologies defined by the SU(3) colour algebra.
- Improves description of baryon production.



ordinary string reconnection



double junction reconnection



triple junction reconnection

more types of junction  
reconnections available..

**Some free parameters:**

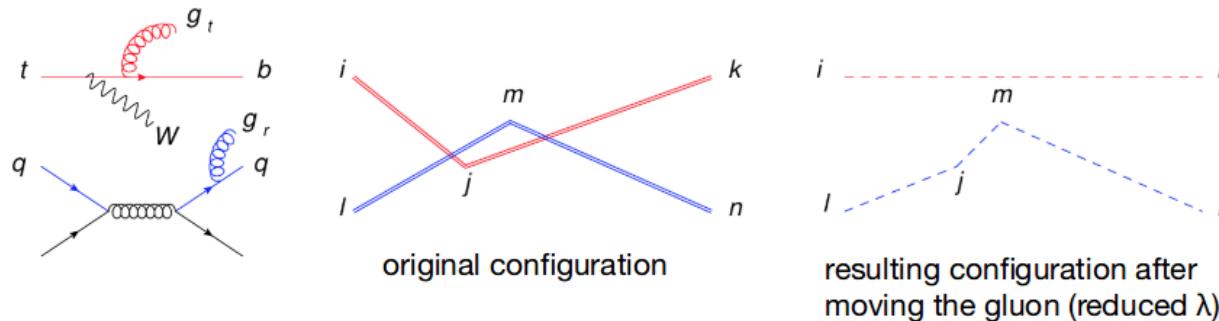
**m0:** variable used in the  $\lambda$  measure for the string length.

**timeDilationPar:** controls the time of two strings to resolve each other between formation and hadronization.

**junctionCorrection:** multiplicative factor to control junction production.

# Gluon move based Color Reconnection

Similar to other CR models, it aims to reduce the total "string length"  $\lambda$ .



→ A gluon  $j$  originally attached to a string piece,  $ik$ , can be moved to a different string piece,  $lm$ , if it leads to a smaller total string length  $\lambda$ .

$$\Delta\lambda(j, lm) = \lambda_{j;lm} - \lambda_{j;ik} = \lambda_{lj} + \lambda_{jm} + \lambda_{ik} - (\lambda_{ij} + \lambda_{jk} + \lambda_{lm}).$$

$$\min_{j,lm} \Delta\lambda(j, lm) \leq \Delta\lambda_{\text{cut}}$$

## Some free parameters:

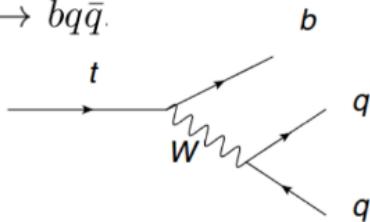
**m2Lambda:** variable used in the definition of  $\lambda$ .

10.1007/JHEP11(2014)043

**fracGluon:** probability that a given gluon will be considered for being moved.

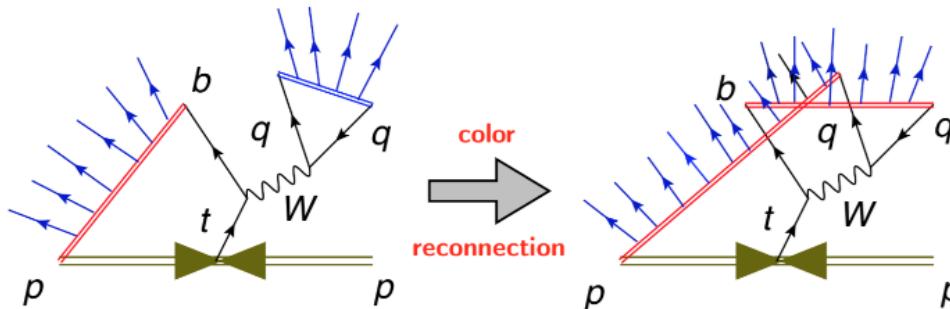
# Color Reconnection on top mass

$$t \rightarrow bW \rightarrow bq\bar{q}$$



$$\hat{m}_{\text{top}}^2 = (p_b + p_{j1} + p_{j2})^2$$

Leakage of hadrons from jet due to CR affects  $m_{\text{top}}$ .



Typical hadronization scale is around 1 fm.

→ But top quark travels ~0.2 fm before it decays.

In PYTHIA8 (Early Resonance Decay (ERD))

- **ERD = off**: top quark can colour reconnect to other partons.
- **ERD = on**: the decay products of the top quark can colour reconnect to other partons.

Credits: Spyros Argyropoulos  
10.1007/JHEP11(2014)043

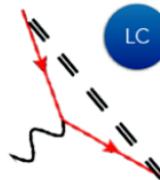
# Color Reconnection on top mass

## Color reconnection (CR)

CR appears to be required to describe soft effects in pp

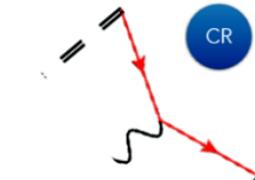
The basic effect on jets is 'string drag'

Simple example:  
Jets from hadronic  
W decay

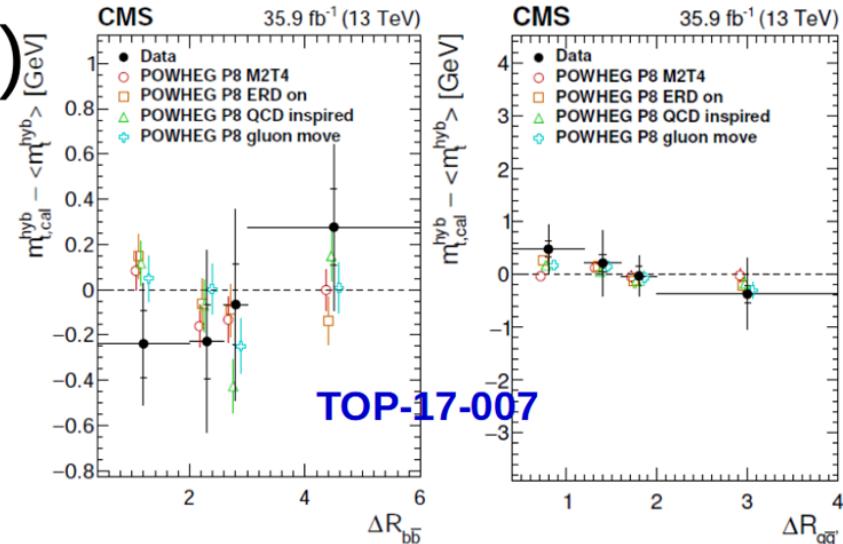


P. Skands

Reconstructed opening angle  
smaller than at parton level



Reconstructed opening angle  
larger than at parton level



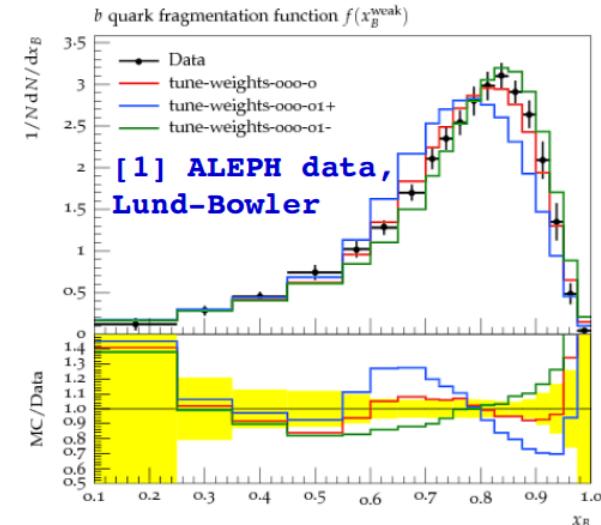
- Pythia models considered for Run 2:
  - **Default:** multiparton interactions (MPI) based → simple (1 free parameter)
  - **QCD based** ([arXiv:1505.01681](https://arxiv.org/abs/1505.01681)): string formation beyond LO
  - **Gluon move** ([arXiv:1407.6653](https://arxiv.org/abs/1407.6653)): allows gluons to be move to another string
  - **Early resonance decays (ERD) on (or off):** color reconnection allowed between top quark decay products and underlying event (or not)
- Recommendation: use the largest effect → ~0.3 GeV for  $m_t$  (**TOP-17-007**)

# b fragmentation

- b quark fragmentation: momentum transferred from B hadron to b jet,  $x_b = p_T(B) / p_T(b \text{ jet})$
- Modeled in Pythia by the Lund-Bowler or Peterson functions, e.g. Lund-Bowler:

$$\frac{1}{z^{1+r_b+b*m_b^2}} * (1-z)^a * \exp\left(-\frac{b*m_T^2}{z}\right)$$

- z is the momentum fraction of the b quark carried by the b hadron,  $m_b$  is the mass of the b quark and  $m_T^2 = m_{\text{had}}^2 + p_{T,\text{had}}^2$
- a and b are general fit parameters;  $r_b$  and  $m_b$  are specific to b quark fragmentation,  $r_b$  determined at LEP by fitting  $x_b$
- Vary  $x_b$  according to uncertainties by reweighting events, also for Peterson function → effect similar to FSR in parton shower
- Usually not a dominant uncertainty



# Flavour response / hadronization

- Jets originating by partons of different flavor (gluons, light quarks, c and b quarks) have a different energy response
- The uncertainty on the energy response is evaluated by
  - Considering the difference in energy response between Pythia 6 and Herwig++ (different hadronization models: Lund string vs cluster)  
→ future: check difference between Pythia 8 and Herwig 7
  - Varying jet energy scales according to the expected flavor composition of the sample

