LUND UNIVERSITY

ACLA Monte Carlo
group meetings, CERN,
Nov – Dec 2007

PYTHIA 8.1

Introduction
and
Tutorial

Torbjörn Sjöstrand

CERN/PH and

Department of Theoretical Physics, Lund University

PYTHIA 6 status

PYTHIA has its roots in JETSET, begun in 1978 → almost 30 years.

PYTHIA 6 still being (slightly) developed and (fully) maintained:

- multiple interactions and underlying event, with
- transverse-momentum-ordered showers
- SUSY interfaces (SLHA) and simulation
- regular bug fixes and minor improvements
- moved to CEDAR HepForge (code management, bugtracking)

Currently PYTHIA 6.413:

- 75,000 lines of code (including comments/blanks)
- 580 page PYTHIA 6.4 Physics and Manual
  T. Sjöstrand, S. Mrenna and P. Skands,
  JHEP**05** (2006) 026 [hep-ph/0603175]
- + update notes, sample main programs, etc.

. . . but

- only add, never subtract

⇒ has become bloated and unmanageable

- is in Fortran 77, so not understood by young people

PYTHIA 8: plans and reality

Tentative schedule (spring 2003):

time	date	processes	final states
0 =	1 Sept. 2004	—	—
1 =	1 Sept. 2005	LHA-style input	incomplete draft
2 =	1 Sept. 2006	a few processes	complete, buggy(?)
3 =	1 Sept. 2007	more processes	stable, debugged

Status: involuntary break ∼6 months + Murphy's law

$\Longrightarrow$ currently ∼ at year 2.5

PYTHIA 8.100 released on 20 October:

- Webpages revamped

  Recent ⇐ PYTHIA 6.4

  Present ⇐ PYTHIA 8.1

  Future ⇐ loose plans
- A Brief Introduction to PYTHIA 8.1

  in arXiv:0710.3820

  submitted to CPC

PYTHIA 8 status

task	status
administative structure	operational; extensions planned
hard processes, internal	much of PYTHIA 6; SUSY & TC & more to do
resonance decays	much of PYTHIA 6; SUSY & TC & more to do
hard processes, external	interfaces to LHA F77, LHEF, PYTHIA 6
SUSY(+more) parameters	primitive SLHA2; more needed
initial-state showers	operational
final-state showers	operational
matching ME’s to showers	some exists; much more needed
multiple interactions	operational; extensions planned
beam remnants & colour flow	operational; alternatives to come
parton densities	only 2 internal, but interface to LHAPDF
string fragmentation	operational; improvements planned
decays & particle data	operational; may need updates
Bose-Einstein	operational; off by default (tuning)
analysis	some simple tools; may be enough
graphical user interface	operational; could be extended
tuning	major task for MCnet postdocs!
testing	major task for experimentalists!

Key differences between PYTHIA 6.4 and 8.1

Old features definitely removed include, among others:

- independent fragmentation
- mass-ordered showers

Features omitted so far include, among others:

- ep, γp and $\gamma\gamma$ beam configurations
- several processes, especially SUSY & Technicolor

New features, not found in 6.4:

- interleaved $p_\perp$-ordered MI + ISR + FSR evolution
- richer mix of underlying-event processes (γ, J/ψ, DY, . . .)
- possibility for two selected hard interactions in same event
- possibility to use one PDF set for hard process and another for rest
- elastic scattering with Coulomb term (optional)
- updated decay data

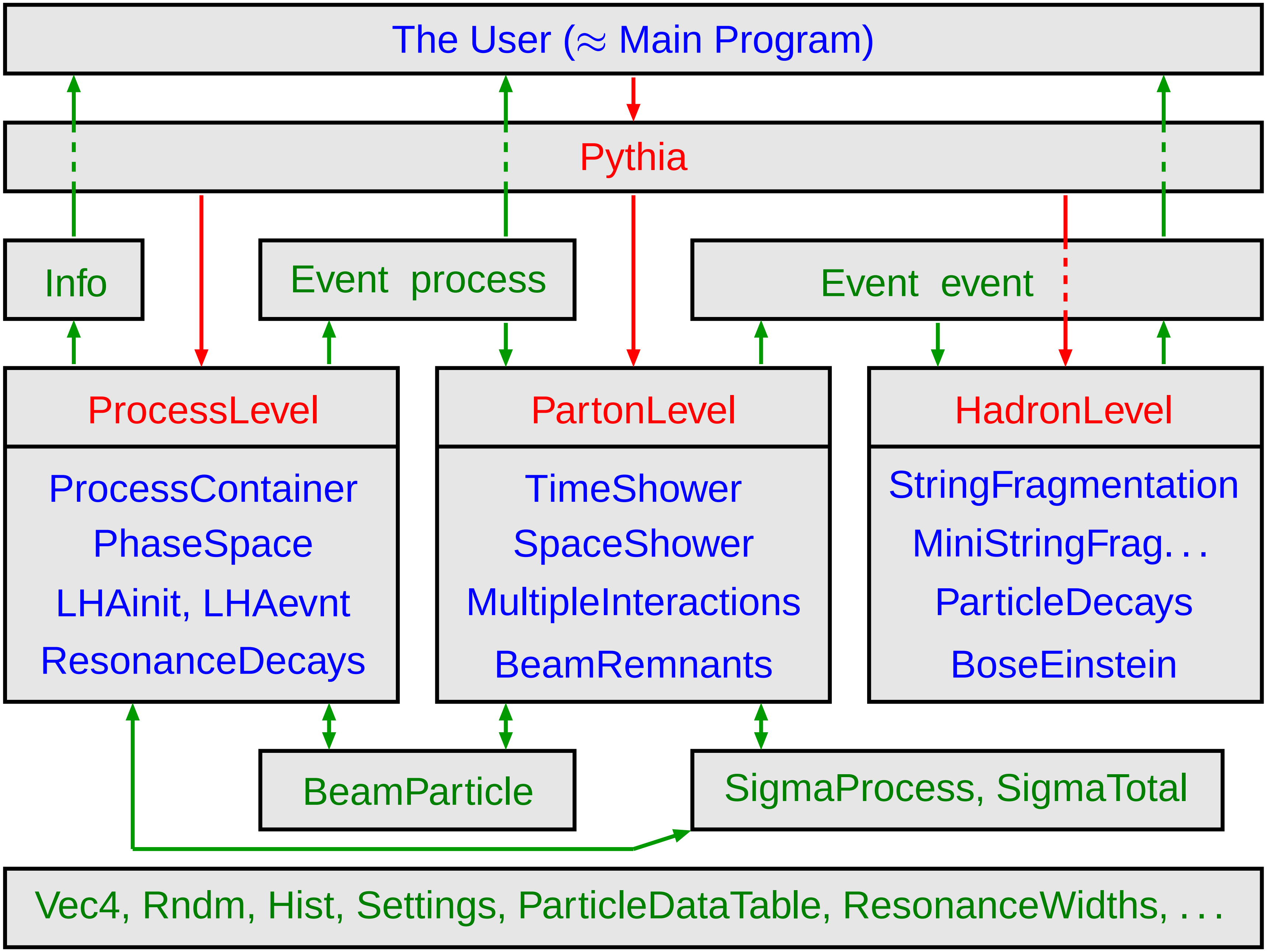
Preliminary plans for the future:

- rescattering in multiple interactions
- NLO and L-CKKW matching

Trying It Out

- Download `pythia8100.tgz` from
  `http://www.thep.lu.se/∼torbjorn/Pythia.html`
- `tar xvfz pythia8100.tgz` to unzip and expand
- `cd pythia8100` to move to new directory
- `./configure ...` needed for external libraries + debug/shared
  (see `README`, libraries: HepMC, LHAPDF, PYTHIA 6)
- `make` will compile in ∼ 3 minutes
  (for archive library, same amount extra for shared)
- The `htmldoc/pythia8100.pdf` file contains A Brief Introduction
- Open `htmldoc/Welcome.html` in a web browser for the full manual
- Install the `phpdoc/` directory on a webserver and open
  `phpdoc/Welcome.html` in a web browser for an interactive manual
- The `examples` subdirectory contains 30 sample main programs:
  standalone, link to libraries, semi-internal processes, . . .
  (`make mainNN` and then `./mainNN.exe > outfile`)
- A `Worksheet` (on the web pages) contains step-by-step
  instructions and exercises how to write and run a main program

PYTHIA 8 structure



The User (≈ Main Program)

Pythia

Info

Event process

Event event

ProcessLevel

ProcessContainer

PhaseSpace

LHAinit, LHAevnt

ResonanceDecays

PartonLevel

TimeShower

SpaceShower

MultipleInteractions

BeamRemnants

HadronLevel

StringFragmentation

MiniStringFrag. . .

ParticleDecays

BoseEinstein

BeamParticle

SigmaProcess, SigmaTotal

Vec4, Rndm, Hist, Settings, ParticleDataTable, ResonanceWidths, . . .

Example of a main program

```
// File: main01.cc. The charged multiplicity distribution at the LHC.
#include "Pythia.h"
using namespace Pythia8;
int main() {
  // Generator. Process selection. LHC initialization. Histogram.
  Pythia pythia;
  pythia.readString("HardQCD:all = on");
  pythia.readString("PhaseSpace:pTHatMin = 20.");
  pythia.init( 2212, 2212, 14000.);
  Hist mult("charged multiplicity", 100, -0.5, 799.5);
  // Begin event loop. Generate event. Skip if error. List first one.
  for (int iEvent = 0; iEvent < 100; ++iEvent) {
    if (!pythia.next()) continue;
    if (iEvent < 1) {pythia.info.list(); pythia.event.list();}
    // Find number of all final charged particles and fill histogram.
    int nCharged = 0;
    for (int i = 0; i < pythia.event.size(); ++i)
      if (pythia.event[i].isFinal() && pythia.event[i].isCharged())
        ++nCharged;
    mult.fill( nCharged );
  // End of event loop. Statistics. Histogram. Done.
  }
  pythia.statistics();
  cout << mult;
  return 0;
}
```

Initialization and generation commands

Standard in beginning:

- `#include "Pythia.h"`
- `using namespace Pythia8;`
- `Pythia pythia;`

Initialization by one of different forms:

- `pythia.init( idA, idB, eA, eB)` along $\pm z$ axis
- `pythia.init( idA, idB, eCM)` in c.m. frame
- `pythia.init( "filename")` for Les Houches Event Files
- `pythia.init()` takes above kinds of input from “cards”
- `pythia.init( LHAinit*, LHAevnt*)` for Les Houches Accord

returns `false` if failed (normally user setup mistake!)

Generation of next event by:

- `pythia.next()`

with no arguments, but value `false` if failed (rare!)

At the end of the generation loop:

- `pythia.statistics()`

provides some summary information

Settings and Particle Data

Can read in settings and particle data changes by

- `pythia.readString("command")`
- `pythia.readFile("filename")` with one `command` per line in file

Settings come in four kinds

- Flags: on/off switches, `bool`
  (`on = yes = ok = true = 1, off = no = false = 0`)
- Modes: enumerated options, `int`
- Parms: (short for parameters) continuum of values, `double`
- Words: characters (no blanks), `string`

and `command` is of form `task:property = value`, e.g.
`PartonLevel:ISR = off` no initial-state radiation
`SigmaProcess:alphaSorder = 0` freeze α_s
`TimeShower:pTmin = 1.0` cut off final-state radiation at 1 GeV

To access **particle data**, instead `command` should be of form
`id:property = value` or `id:channel:property = value`, e.g.
`3122:mayDecay = no` do not allow Λ^0 to decay
`215:3:products = 211 111 111` to let $a_2^+ \to \pi^+\pi^0\pi^0$

Note: case-insensitive search/matching in databases!

Example of a “cards” file

```
! This file contains commands to be read in for a Pythia8 run.
! Lines not beginning with a letter or digit are comments.

! 1) Settings that could be used in a main program, if desired.
Main:idBeamA = 2212                ! first beam, p = 2212, pbar = -2212
Main:idBeamB = 2212                ! second beam, p = 2212, pbar = -2212
Main:eCM = 14000.                  ! CM energy of collision
Main:numberOfEvents = 1000         ! number of events to generate
Main:numberToList = 2              ! number of events to print
Main:timesToShow = 20              ! show how far along run is
Main:showChangedSettings = on      ! print changed flags/modes/parameters
Main:showAllSettings = off         ! print all flags/modes/parameters

! 2) Settings for the hard-process generation.
HiggsSM:gg2H = on                  ! Higgs production by gluon-gluon fusion
25:m0 = 123.5                      ! Higgs mass
25:onMode = off                    ! switch off all Higgs decay channels
25:onIfMatch = 22 22               ! switch back on Higgs -> gamma gamma
SigmaProcess:alphaSvalue = 0.12    ! alpha_s(m_Z) in matrix elements

! 3) Settings for the subsequent event generation process.
SpaceShower:alphaSvalue = 0.13     ! alpha_s(m_Z) in initial-state radiation
MultipleInteractions:pT0Ref = 3.0  ! pT_0 regularization at reference energy
#PartonLevel:MI = off               ! no multiple interactions
#PartonLevel:ISR = off              ! no initial-state radiation
#PartonLevel:FSR = off              ! no final-state radiation
#HadronLevel:Hadronize = off        ! no hadronization
```

More on settings

Settings are stored in four separate maps (flags/modes/parms/words).
For each setting, need to store

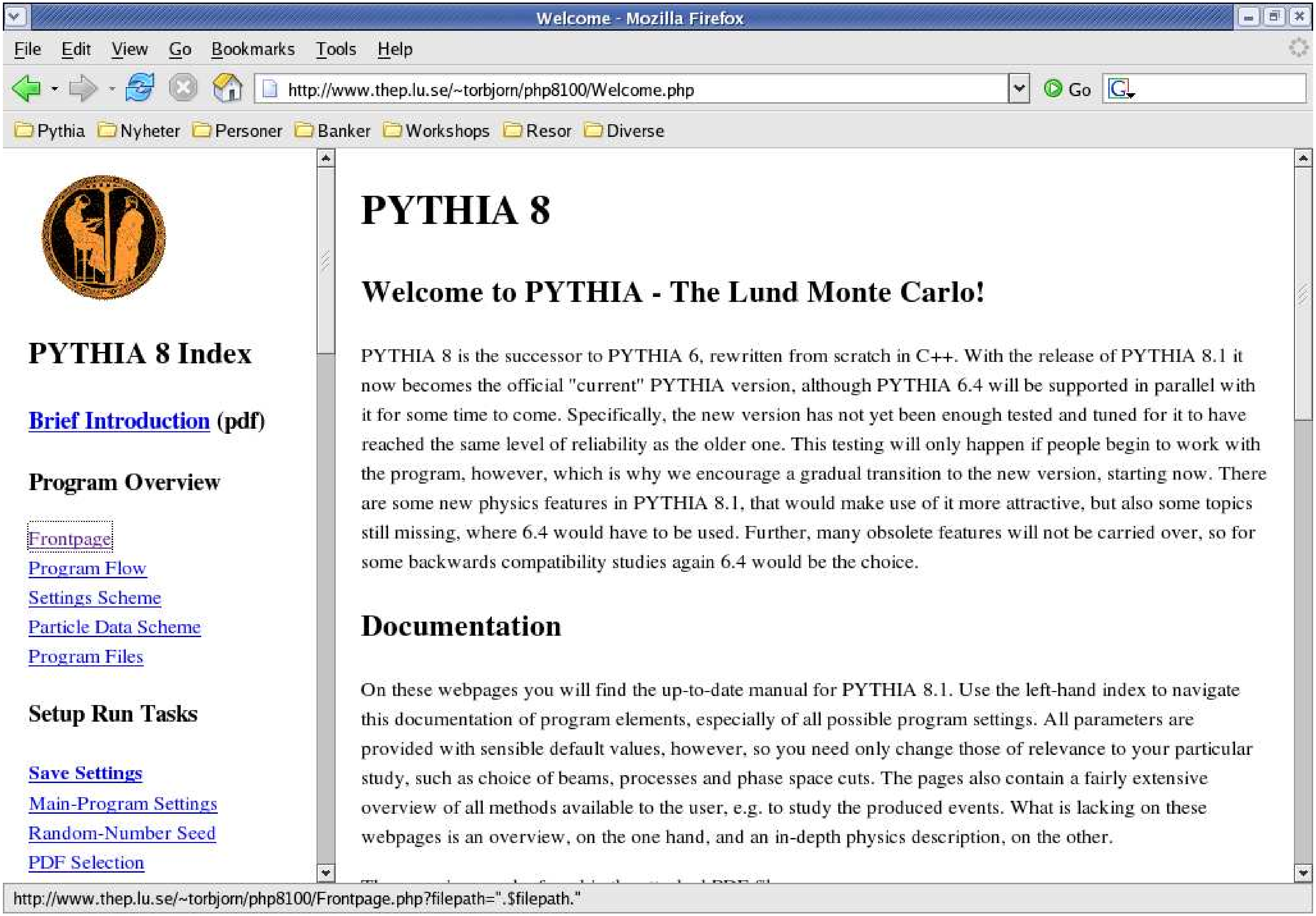
- name: of form `task:property`, e.g. `TimeShower:pTmin`
- default value
- current value
- allowed range: minimum/maximum on/off (not for flags).

Useful commands:

- `pythia.settings.listAll()` : complete list
- `pythia.settings.listChanged()` : only changed ones

```
*-------  PYTHIA Flag + Mode + Parm + Word Settings (changes only)  --------------------------*
|                                                                                             |
| Name                                    |          Now |      Default         Min          Max |
|                                         |              |                                       |
| HardQCD:all                             |           on |          off                            |
| Main:eCM                                |    14000.000 |     1960.000    10.00000              |
| Main:numberToList                       |            1 |            2           0              |
| Main:showChangedParticleData            |           on |          off                            |
| Main:timesToShow                        |           20 |           50           0              |
| MultipleInteractions:pTmin              |      3.00000 |      0.20000     0.10000     10.00000 |
| PhaseSpace:pTHatMin                     |     50.00000 |          0.0         0.0              |
| PromptPhoton:all                        |           on |          off                            |
| SpaceShower:pT0Ref                      |      2.00000 |      2.20000     0.50000     10.00000 |
|                                                                                             |
*-------  End PYTHIA Flag + Mode + Parm + Word Settings  ------------------------------------*
```

Online manual $\Longrightarrow$ Graphical User Interface


Welcome - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://www.thep.lu.se/~torbjorn/php8100/Welcome.php
Go
Pythia Nyheter Personer Banker Workshops Resor Diverse
http://www.thep.lu.se/~torbjorn/php8100/Frontpage.php?filepath=".$filepath."

PYTHIA 8 Index

Brief Introduction (pdf)

Program Overview

Frontpage
Program Flow
Settings Scheme
Particle Data Scheme
Program Files

Setup Run Tasks

Save Settings
Main-Program Settings
Random-Number Seed
PDF Selection

PYTHIA 8

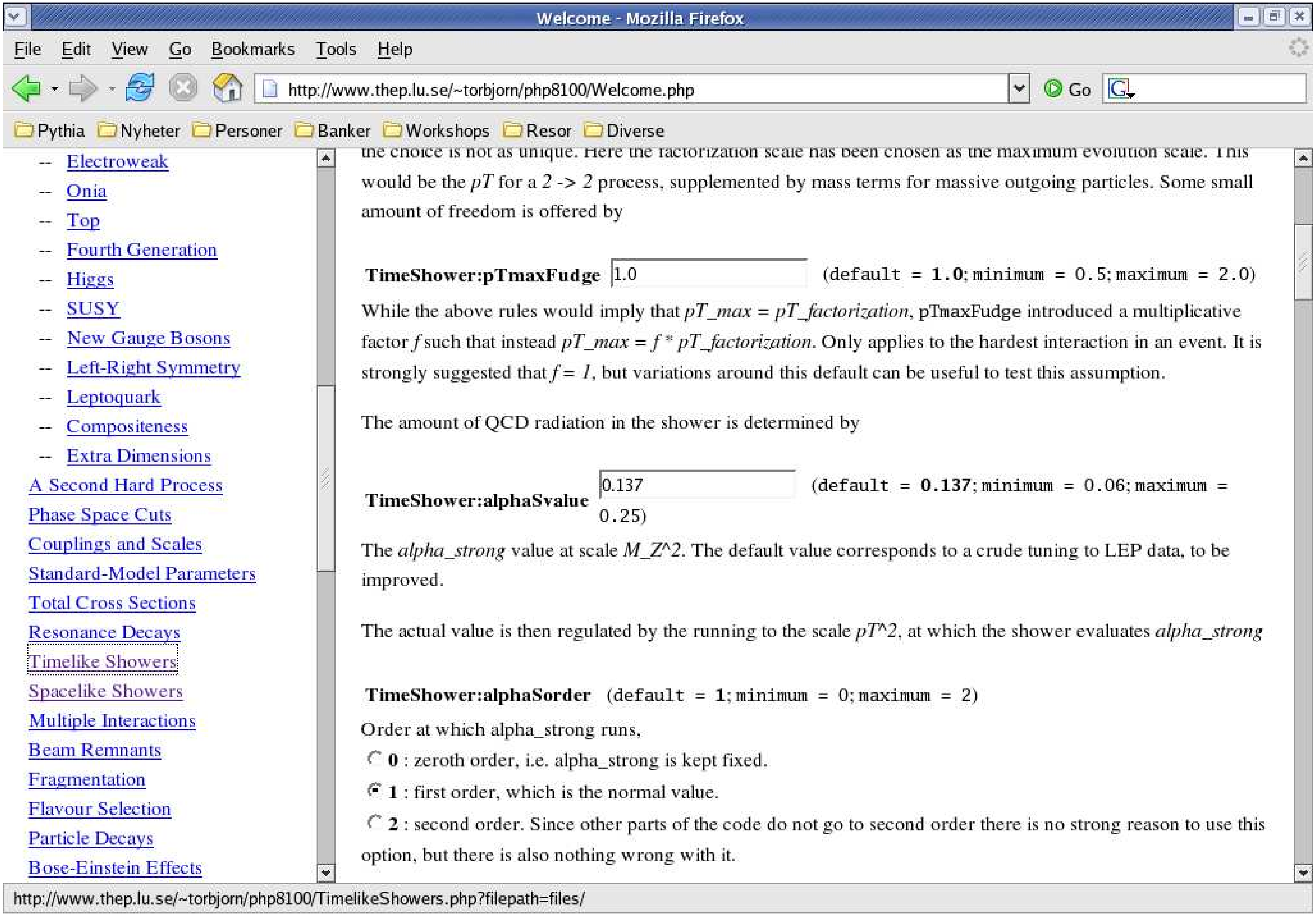
Welcome to PYTHIA - The Lund Monte Carlo!

PYTHIA 8 is the successor to PYTHIA 6, rewritten from scratch in C++. With the release of PYTHIA 8.1 it now becomes the official "current" PYTHIA version, although PYTHIA 6.4 will be supported in parallel with it for some time to come. Specifically, the new version has not yet been enough tested and tuned for it to have reached the same level of reliability as the older one. This testing will only happen if people begin to work with the program, however, which is why we encourage a gradual transition to the new version, starting now. There are some new physics features in PYTHIA 8.1, that would make use of it more attractive, but also some topics still missing, where 6.4 would have to be used. Further, many obsolete features will not be carried over, so for some backwards compatibility studies again 6.4 would be the choice.

Documentation

On these webpages you will find the up-to-date manual for PYTHIA 8.1. Use the left-hand index to navigate this documentation of program elements, especially of all possible program settings. All parameters are provided with sensible default values, however, so you need only change those of relevance to your particular study, such as choice of beams, processes and phase space cuts. The pages also contain a fairly extensive overview of all methods available to the user, e.g. to study the produced events. What is lacking on these webpages is an overview, on the one hand, and an in-depth physics description, on the other.

Example: timelike parton showers


Welcome - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://www.thep.lu.se/~torbjorn/php8100/Welcome.php
Go
Pythia Nyheter Personer Banker Workshops Resor Diverse

the choice is not as unique. Here the factorization scale has been chosen as the maximum evolution scale. This would be the *pT* for a *2 -> 2* process, supplemented by mass terms for massive outgoing particles. Some small amount of freedom is offered by

TimeShower:pTmaxFudge 1.0 (default = **1.0**; minimum = 0.5; maximum = 2.0)

While the above rules would imply that *pT_max = pT_factorization*, pTmaxFudge introduced a multiplicative factor *f* such that instead *pT_max = f * pT_factorization*. Only applies to the hardest interaction in an event. It is strongly suggested that *f = 1*, but variations around this default can be useful to test this assumption.

The amount of QCD radiation in the shower is determined by

TimeShower:alphaSvalue 0.137 (default = **0.137**; minimum = 0.06; maximum = 0.25)

The *alpha_strong* value at scale *M_Z^2*. The default value corresponds to a crude tuning to LEP data, to be improved.

The actual value is then regulated by the running to the scale *pT^2*, at which the shower evaluates *alpha_strong*

TimeShower:alphaSorder (default = **1**; minimum = 0; maximum = 2)

Order at which alpha_strong runs,

○ **0** : zeroth order, i.e. alpha_strong is kept fixed.

● **1** : first order, which is the normal value.

○ **2** : second order. Since other parts of the code do not go to second order there is no strong reason to use this option, but there is also nothing wrong with it.

http://www.thep.lu.se/~torbjorn/php8100/TimelikeShowers.php?filepath=files/

Manual Sections

Program Overview

Frontpage
Program Flow
Settings Scheme
Particle Data Scheme
Program Files

Setup Run Tasks

Save Settings
Main-Program Settings
Random-Number Seed
PDF Selection
Master Switches
Process Selection
– QCD
– Electroweak
– Onia
– Top
– Fourth Generation
– Higgs
– SUSY
– New Gauge Bosons
– Left-Right Symmetry
– Leptoquark
– Compositeness
– Extra Dimensions
A Second Hard Process
Phase Space Cuts
Couplings and Scales
Standard-Model Parameters
Total Cross Sections
Resonance Decays
Timelike Showers
Spacelike Showers
Multiple Interactions
Beam Remnants
Fragmentation
Flavour Selection
Particle Decays
Bose-Einstein Effects
Particle Data
Error Checks
Tunes

Study Output

Four-Vectors
Particle Properties
Event Record
Event Information
Event Statistics
Histograms
Event Analysis
HepMC Interface

Link to Other Programs

Les Houches Accord
Access PYTHIA 6 Processes
Semi-Internal Processes
Semi-Internal Resonances
Hadron-Level Standalone
SUSY Les Houches Accord
Parton Distributions
External Decays
User Hooks
Random Numbers
Implement New Showers

Reference Materiel

Bibliography
Glossary

Version

Hard-process generation

Processes can be switched on with

```
ProcessGroup:ProcessName = on
```

or sometimes

```
ProcessGroup:all = on
```

ProcessGroup	ProcessName
SoftQCD	minBias,elastic, singleDiffractive, doubleDiffractive
HardQCD	gg2gg, gg2qqbar, qg2qg, qq2qq, qqbar2gg, qqbar2qqbarNew, gg2ccbar, qqbar2ccbar, gg2bbbar, qqbar2bbbar
PromptPhoton	qg2qgamma, qqbar2ggamma, gg2ggamma, ffbar2gammagamma, gg2gammagamma
WeakBosonExchange	ff2ff(t:gmZ), ff2ff(t:W)
WeakSingleBoson	ffbar2gmZ, ffbar2W, ffbar2ffbar(s:gm)
WeakDoubleBoson	ffbar2gmZgmZ, ffbar2ZW, ffbar2WW
WeakBosonAndParton	qqbar2gmZg, qg2gmZq, ffbar2gmZgm, fgm2gmZf qqbar2Wg, qg2Wq, ffbar2Wgm, fgm2Wf
Charmonium	gg2QQbar[3S1(1)]g, qg2QQbar[3PJ(8)]q, ...
Bottomonium	gg2QQbar[3S1(1)]g, gg2QQbar[3P2(1)]g, ...

ProcessGroup	ProcessName
Top	gg2ttbar, qqbar2ttbar, qq2tq(t:W), ffbar2ttbar(s:gmZ), ffbar2tqbar(s:W)
FourthBottom	gg2bPrimebPrimebar, qq2bPrimeq(t:W) , ...
FourthTop	qqbar2tPrimetPrimebar, fbar2tPrimeqbar(s:W), ...
FourthPair	ffbar2tPrimebPrimebar(s:W), fbar2tauPrimenuPrimebar(s:W)
HiggsSM	ffbar2H, gg2H, ffbar2HZ, ff2Hff(t:WW), ...
HiggsBSM	h, H and A as above, charged Higgs, pairs
SUSY	qqbar2chi0chi0 (SUSY barely begun)
NewGaugeBoson	ffbar2gmZZprime, ffbar2Wprime, ffbar2R0
LeftRightSymmmetry	ffbar2ZR, ffbar2WR, ffbar2HLHL, ...
LeptoQuark	ql2LQ, qg2LQl, gg2LQLQbar, qqbar2LQLQbar
ExcitedFermion	dg2dStar, qq2uStarq, qqbar2muStarmu, ...
ExtraDimensionsG*	gg2G*, qqbar2G*, ...

Can also use (and sometimes mix with)

- Les Houches Event Files
- Les Houches Accord-style runtime C++ interface
- Les Houches Accord runtime Fortran 77 interface
  (and that way runtime link to PYTHIA 6.4)
- semi-internal matrix elements and resonances
  (external matrix elements, internal phase space)

More on particle data

The static `ParticleDataTable` class contains info by PDG `id` code:

- `name(id), hasAnti(id)`
- `spinType(id), chargeType(id), charge(id), colType(id)`
- `m0(id), mWidth(id), mMin(id), mMax(id), tau0(id), ...`

plus a vector of `DecayChannel`s with

- `onMode(), bRatio(), meMode(), multiplicity(), product(i)`

User modifies by methods, `readString("...")` and `readFile("filename")` with commands `id:property = value` or `id:channel:property = value`.

Some special commands:

```
id:all = name antiName spinType chargeType colType m0 mWidth mMin mMax tau0
id:new = name antiName spinType chargeType colType m0 mWidth mMin mMax tau0
id:channel:all = onMode bRatio meMode products
id:oneChannel = onMode bRatio meMode products
id:addChannel = onMode bRatio meMode products
id:onMode = onMode
id:onIfAny = products    and   id:offIfAny = products
id:onIfAll = products    and   id:offIfAll = products
id:onIfMatch = products  and  id:offIfMatch = products
```

Useful commands:

- `pythia.particleData.listAll()` : complete list
- `pythia.particleData.listChanged()` : only changed ones
- `pythia.particleData.list(id)` : only one (or `vector<int>`)

```
 --------  PYTHIA Particle Data Table (changed only)  ---------------------------
----------------------------------------------------


      id   name            antiName         spn chg col      m0         mWidth
   mMin       mMax       tau0    res dec ext vis wid
             no onMode   bRatio    meMode     products

     111  pi0                                 1   0   0    0.13498    0.00000
 0.00000    0.00000  2.51000e-05   0   0   0   1   0
              0     1   0.9879900    0       22       22
              1     1   0.0119800   11       22       11      -11
              2     1   0.0000300   13       11      -11       11      -11

     223  omega                               3   0   0    0.78259    0.00849
 0.10000    0.00000  0.00000e+00   0   1   0   1   0
              0     1   0.8924000    1      211     -211      111
              1     1   0.0892800    0       22      111
              2     1   0.0170000    3      211     -211
              3     1   0.0004900    0      221       22
              4     1   0.0000700    0      111      111       22
              5     1   0.0005900    0      111       11      -11
              6     1   0.0001000    0      111       13      -13
              7     1   0.0000700    0       11      -11


 --------  End PYTHIA Particle Data Table  --------------------------------------
----------------------------------------------------
```

The Particle class in the event record

Each `Particle` object stores the properties:

- `id()` : particle identity, by PDG codes.
- `status()` : status code. Provides info on where and why a given particle was produced. Negative code = no longer existing particle.
- `mother1(), mother2()` : first and last mother indices.
- `daughter1(), daughter2()` : first and last daughter indices.
- `col(), acol()` : colour and anticolour tags, Les Houches Accord.
- `px(), py(), pz(), e()` : four-momentum components (in GeV).
- `m()` : mass.
- `scale()` : scale at which a parton was produced; model-specific.
- `xProd(), yProd(), zProd(), tProd()` : production vertex (in mm).
- `tau()` : proper lifetime.

Methods above can also be used, with argument, for setting properties.
Many further methods for extraction only, e.g. for rapidity.
Also pointer to `ParticleDataTable` entry; gives e.g. `name()` and `charge()`.

The Event class

Two `Event` objects inside a `Pythia` object:

- `process` : hard subprocess, roughly like Les Houches.
- `event` : complete event history.

$\boxed{\text{An } \texttt{Event} \approx \text{a } \texttt{vector<Particle>}}$

e.g. `pythia.event[i].id()` = identity of i’th particle

index 0 = event-as-a-whole; not really part of history

- ⇒ throw line 0 for HepMC conversion
- ⇒ mother/daughter = 0 ⇔ empty

Specific methods include:

- `size()` : 0 ≤`i`< `event.size()`.
- `list()` : provide event listing.
- `motherList(i), daughterList(i), sisterList()` :
  a `vector<int>` of mothers, daughters, sisters.
- `iTopCopy(i), iBotCopy(i)` : top or bottom “carbon copy”.

But *no* methods to edit the event.

Further: info on junctions, subsystems (multiple interactions), . . .

Sample event listings

First with `pythia.process.list()`, truncated to fit:

```
 --------  PYTHIA Event Listing  (hard process)  -----------------------------------------

    no        id   name            status     mothers   daughters     colours      p_x
     0        90   (system)           -11     0     0     1     2     0     0      0.000
     1      2212   (p+)               -12     0     0     3     0     0     0      0.000
     2      2212   (p+)               -12     0     0     4     0     0     0      0.000
     3        -2   (ubar)             -21     1     0     5     6     0   101      0.000
     4         2   (u)                -21     2     0     5     6   102     0      0.000
     5        -6   (tbar)             -22     3     4     7     8     0   101    -73.897
     6         6   (t)                -22     3     4     9    10   102     0     73.897
     7       -24   (W-)               -22     5     0    11    12     0     0      2.825
     8        -5   bbar                23     5     0     0     0     0   101    -76.721
     9        24   (W+)               -22     6     0    13    14     0     0     72.384
    10         5   b                   23     6     0     0     0   102     0      1.513
    11         3   s                   23     7     0     0     0   103     0    -26.914
    12        -4   cbar                23     7     0     0     0     0   103     29.739
    13       -11   e+                  23     9     0     0     0     0     0      6.458
    14        12   nu_e                23     9     0     0     0     0     0     65.926
                                Charge sum:   0.000           Momentum sum:        0.000

 --------  End PYTHIA Event Listing  ---------------------------------------------------
```

next with `pythia.event.list()`, omissions to fit:

```
 --------  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     1     2     0     0      0.000      0.000      0.000  14000.000  14000.000
     1      2212   (p+)               -12     0     0   279     0     0     0      0.000      0.000   7000.000   7000.000      0.938
     2      2212   (p+)               -12     0     0   280     0     0     0      0.000      0.000  -7000.000   7000.000      0.938
     3        -2   (ubar)             -21     7     7     5     6     0   101      0.000      0.000     54.594     54.594      0.000
     4         2   (u)                -21     8     0     5     6   102     0      0.000      0.000  -1042.471   1042.471      0.000
     5        -6   (tbar)             -22     3     4     9     9     0   101    -73.897    -53.244   -174.768    261.166    171.372
     6         6   (t)                -22     3     4    10    10   102     0     73.897     53.244   -813.108    835.899    171.131
     7        -2   (ubar)             -42    12     0     3     3     0   101      0.000      0.000     54.594     54.594      0.000
     8         2   (u)                -41    13    13    11     4   104     0     -0.000     -0.000  -1191.549   1191.549      0.000
     9        -6   (tbar)             -44     5     5    14    14     0   101    -71.565    -51.768   -210.234    285.251    171.372
    10         6   (t)                -44     6     6    15    15   102     0     82.715     58.828   -926.573    947.695    171.131
    11        21   (g)                -43     8     0    16    16   104   102    -11.150     -7.060     -0.149     13.198      0.000
    25        21   (g)                -51    23     0    37    37   106   105     19.037     28.329     38.331     51.325      0.000
    26        21   (g)                -51    23     0    39    39   101   106      6.832    -19.532      2.861     20.889      0.000
    27        -6   (tbar)             -52    20    20    34    34     0   101    -88.187    -52.597   -231.302    305.635    171.372
    44        21   (g)                -31    48     0    46    47   114   113      0.000      0.000      0.707      0.707      0.000
    45         1   (d)                -31    49    49    46    47   113     0      0.000      0.000   -255.118    255.118      0.000
    46        21   (g)                -33    44    45    50    50   114   115      2.524      5.061    -11.187     12.535      0.000
    47         1   (d)                -33    44    45    51    51   115     0     -2.524     -5.061   -243.224    243.290      0.330
   378         2   (u)                -63     1     0   492   492   113     0     -0.319     -0.512   1340.638   1340.638      0.330
   379      2101   (ud_0)             -63     1     0   492   492     0   113     -0.427     -1.024   3266.905   3266.906      0.579
   380         2   (u)                -63     1     0   493   493   108     0     -0.720     -1.118     56.936     56.952      0.330
   381        -3   (sbar)             -63     1     0   519   519     0   117     -0.382     -0.112   1364.384   1364.384      0.500
   486       -11   e+                  23   441     0     0     0     0     0      7.949    -14.875   -217.791    218.443      0.001
   487        12   nu_e                23   441     0     0     0     0     0     70.533     75.395   -668.054    675.985      0.000
   502         1   (d)                -71   342   342   505   508   115     0     -3.404     -4.046   -233.825    233.885      0.330
   503        21   (g)                -71   367   367   505   508   181   115     -0.384     -0.368     -9.293      9.309      0.000
   504        -2   (ubar)             -71   370   370   505   508     0   181     -3.167     -0.517    -68.782     68.858      0.330
   505       311   (K0)               -83   502   504   789   789     0     0     -2.046     -0.406    -58.420     58.460      0.498
   506       331   (eta')             -83   502   504   941   942     0     0     -1.070     -2.000    -93.597     93.629      0.958
   507      -323   (K*-)              -83   502   504   790   791     0     0     -2.736     -2.575   -132.287    132.344      0.943
   508       111   (pi0)              -84   502   504   943   944     0     0     -1.102      0.050    -27.596     27.618      0.135
   789       130   K_L0                91   505   505     0     0     0     0     -2.046     -0.406    -58.420     58.460      0.498
   790      -311   (Kbar0)            -91   507     0   932   932     0     0     -0.900     -1.003    -55.248     55.267      0.498
   791      -211   pi-                 91   507     0     0     0     0     0     -1.836     -1.571    -77.039     77.077      0.140
   792      -211   pi-                 91   516     0     0     0     0     0      0.117     -0.161     -1.617      1.635      0.140
   793       111   (pi0)              -91   516     0  1069  1070     0     0     -0.431     -0.098     -0.498      0.680      0.135
   794      2212   p+                  91   537     0     0     0     0     0     -1.175      0.093     -0.721      1.670      0.938
   795       211   pi+                 91   537     0     0     0     0     0     -0.414      0.352     -0.340      0.657      0.140
  1316        22   gamma               91  1313     0     0     0     0     0     -1.574      0.014     -0.839      1.783      0.000
  1317        22   gamma               91  1313     0     0     0     0     0     -0.887      0.068     -0.569      1.056      0.000
                                   Charge sum:  2.000           Momentum sum:     -0.000      0.000     -0.000  14000.000  14000.000

 --------  End PYTHIA Event Listing  -----------------------------------------------------------------------------------------------
```

Other event information

You can use `pythia.info.method()` to extract one-of-a-kind information, such as:

- `idA(), idB(), eCM()` : incoming beams and cm energy.
- `name(), code()` : the name and code of the subprocess.
- `id1(), id2(), x1(), x2()` : the identities and x fractions of the two partons coming in to the hard subprocess.
- `pdf1(), pdf2(), Q2Fac()` : parton densities $x\, f(x, Q^2)$ evaluated for the two incoming partons, and the associated Q^2 scale.
- `alphaS(), alphaEM(), Q2Ren()` : α_s, α_{em} and their Q^2 scale.
- `mHat(), sHat(), tHat(), uHat()` : the invariant mass of the hard subprocess and the Mandelstam variables.
- `pTHat(), thetaHat(), phiHat()` : transverse momentum and polar and azimuthal scattering angles of the hard subprocess.
- `bMI(), nMI()` : impact parameter (rescaled) and number of multiple interactions.
- `list()` : list some information on output.
- `sigmaGen(), sigmaErr()` : the process-summed estimated cross section and its estimated statistical error, in mb.

Event analysis

Four-vectors in a class `Vec4`, with overloaded operators.

A small package for one-dimensional histograms:

- Book with `Hist name( title, numberOfBins, xMin, xMax);`
  or `Hist name; name.book(title, numberOfBins, xMin, xMax);`
- Fill with `name.fill( xValue, weight);` with default `weight = 1`
- Print with `cout << name;`
- Overloaded operators for addition, multiplication, . . .

Sphericity analysis (similarly thrust):

- Instantiate with `Sphericity sph( power, select);`
- Analyze with `sph.analyze( event);`
- Info with `sph.sph(), sph.EigenVector(i), sph.list(), ...`

Cone jet finder a la UA1 (PYCELL) (similarly Lund/JADE/Durham):

- Instantiate with `CellJet cellJet( etaMax, nEta, nPhi, select, smear, resolution, upperCut, threshold);`
- Analyze with `cellJet.analyze(event, eTjetMin, coneRadius, eTseed);`
- Info with `cellJet.size(), cellJet.eT(i), cellJet.list(), ...`

Statistics

Output from `pythia.statistics()` (some blanks removed for space):

```
*-------  PYTHIA Event and Cross Section Statistics  -------------------------------------*
|                                                                                         |
| Subprocess                 Code |            Number of events      |     sigma +- delta    |
|                                 |       Tried    Selected   Accepted |    (estimated) (mb)    |
|                                 |                                  |                       |
|-----------------------------------------------------------------------------------------|
|                                 |                                  |                       |
| g g -> g g                  111 |        502          65         65 |  5.114e-01  3.247e-02 |
| g g -> q qbar (uds)         112 |          2           0          0 |  0.000e+00  0.000e+00 |
| q g -> q g                  113 |        247          34         34 |  3.038e-01  2.772e-02 |
| q q(bar)' -> q q(bar)'      114 |         24           0          0 |  0.000e+00  0.000e+00 |
| q qbar -> g g               115 |          1           0          0 |  0.000e+00  0.000e+00 |
| q qbar -> q' qbar' (uds)    116 |          0           0          0 |  0.000e+00  0.000e+00 |
| g g -> c cbar               121 |          1           1          1 |  3.483e-03  3.483e-03 |
| g g -> b bbar               123 |          2           0          0 |  0.000e+00  0.000e+00 |
|                                 |                                  |                       |
| sum                             |        779         100        100 |  8.187e-01  4.284e-02 |
|                                                                                         |
*-------  End PYTHIA Event and Cross Section Statistics ----------------------------------*

*-------  PYTHIA Error and Warning Messages Statistics  ----------------------------------*
|                                                                                         |
|  times   message                                                                        |
|                                                                                         |
|      3   Error in Pythia::next: hadronLevel failed; try again                            |
|      3   Error in StringFragmentation::fragmentToJunction: caught in junction flavour loop |
|      3   Warning in ParticleDataEntry::initBWmass: switching off width                   |
|                                                                                         |
*-------  End PYTHIA Error and Warning Messages Statistics  ------------------------------*
```

Link to other program

PYTHIA is standalone, but several possibilities to link to it.

Posibilities similar to PYTHIA 6.4:

- Input from Les Houches Accord & Les Houches Event Files
- Output to HepMC event format (more robust than PYTHIA 6!?)
- SUSY Les Houches Accord (input file with masses, couplings, . . .)
- Link to external decays, e.g. for τ and B.
- Link to LHAPDF version 5.3.0 or later, or to your own PDF.

New possibilities, based on derived classes and pointers to them:

- Semi-internal process: write derived matrix-element class,

  ```
  SigmaProcess* mySigma = new MySigma();
  pythia.setSigmaPtr( mySigma);
  ```

  and let PYTHIA do phase space integration, process mixing, . . .
- Semi-internal resonance in same style: calculate partial widths
- Link to external random-number generator.
- Link to external shower, e.g. VINCIA for FSR.
- User hooks: veto events early on or reweight cross section.

Sample Main Programs

- `main01.cc`: charged multiplicity distribution
- `main02.cc`: Z^0 $p_\perp$ spectrum
- `main03.cc & main03.cmnd`: single-particle analysis in jet events
- `main04.cc & main04.cmnd`: tests of event properties
- `main05.cc`: cone-jet analysis of LHC events
- `main06.cc & main06.cmnd`: study elastic/diffractive events
- `main07.cc & main07.cmnd`: study minimum-bias events
- `main08.cc & main08.cmnd`: combine results of subruns in $p_\perp$ bins
- `main09.cc`: LEP events with sphericity/thrust/jetfinder analysis
- `main10.cc`: use UserHooks to interact with generation process
- `main11.cc`: set two hard interactions in the same event
- `main12.cc & ttbar.lhe`: input from a Les Houches Event File
- `main13.cc & ttbar.lhe & ttbar2.lhe`: input from two Les Houches Event Files; mix with internal processes
- `main14.cc`: **compare** several cross sections with PYTHIA 6.4 values
- `main15.cc`: redo B decays several times for each event

- `main16.cc`: user analysis class; command-line input file
- `main17.cc`: Pythia wrapper class; command-line input file
- `main21.cc`: input of parton configurations for hadronization only
- `main22.cc & main22.cmnd & main22.spc`: SUSY with SLHA input
- `main23.cc`: link an external decay handler
- `main24.cc`: link an external random number generator
- `main25.cc`: link an external process for internal use
- `main26.cc`: link an external resonance and process for internal use
- `main31.cc & main31.cmnd`: simple output to HepMC event file
- `main32.cc & main32.cmnd`: streamlined production to HepMC; command-line input and output files
- `main41.cc`: test shapes of PDF's in LHAPDF
- `main42.cc`: compare event properties for different LHAPDF PDF's
- `main51.cc`: runtime LHA link to PYTHIA 6.4
- `main52.cc & main52.ccmnd & main52.fcmnd`: ditto with input files
- `main53.f`: (Fortran!) have PYTHIA 6.4 generate an LHEF
- `main54.cc & main54.cmnd`: input from PYTHIA 6.4 and output to HepMC

License and Acknowledgements

Based on MCnet discussions during the spring there is a HERWIG++/SHERPA/PYTHIA/THEPEG/ARIADNE/. . . agreement:

- Our programs are licensed under the GPL version 2.
- Please respect the MCnet Guidelines for Event Generator Authors and Users.
  1. The integrity of the program should be respected.
     - report bugs & fixes to authors — don't create own forks
     - redistribute a program in its entirety, not piecemeal
  2. The program and its physics should be properly cited when used for academic publications.
     - cite manuals, but also physics articles of special relevance
     - cite all programs used, commensurate with importance for study
     - document version/parameters for reproducibility of publications

---

Makefiles, configure scripts & HepMC interface by Mikhail Kirsanov.
Conversion to PHP files by Ben Lloyd.
Win32/NMAKE by Bertrand Bellenot.
Extended Higgs sector by Marc Montull.
Some c/b decay tables from LHCb & DELPHI.

Outlook

We are now in a chicken-and-egg situation:
the user community needs a mature program;
but PYTHIA 8 will only mature
if there is an active user community

So please . . .

- implement in your experimental frameworks
- find volunteers to act as guinea pigs
- do some small-scale “production runs”
- report back problems & wishes (within reason)

Don’t throw away PYTHIA 6.4 just yet!

- 8.1 still can’t do everything 6.4 can
- 8.1 still needs testing and tuning

As new features are introduced, 8.1 will become the obvious choice:

- improved multiple interactions
- more matrix-element matching
- ???