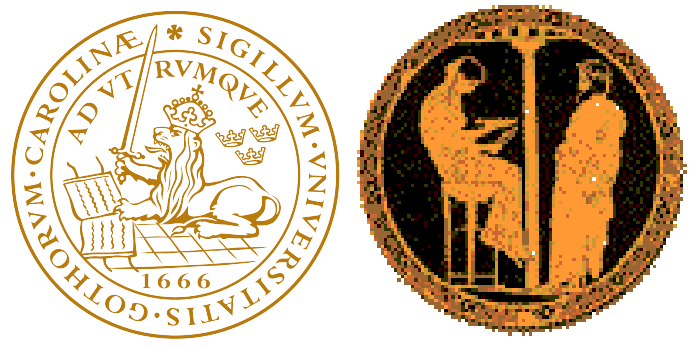
Pythia 8 online tutorial, EIC user group

EIC user group tutorial week, August 17, 2020

Stefan Prestel (Lund)



Welcome!

This is a two-hour long tutorial for new users of Pythia 8.
The goal is to equip you with

- an understanding of the aim of general-purpose event generators, and what "events" look like
- some examples of using Pythia from Python/Jupyter
- a crash course in how to run Pythia from terminal
- and example of using an external event analysis tool

Plan for the next two hours

The next two hours will be split into "lecture" and "study" parts:

1. Introduction lecture (20 min)
   ...basic resources, first runs from Jupyter.
2. Group study (15 min)
   ...look at events at different colliders
3. Pythia terminal steering crash course lecture (40 min)
   ...developing a new main program.
   ...look at simple plots and understand different phenomena.
4. Group study (20 min)
   ...study intrinsic kT and showers at HERA and HERMES.
5. Using external analysis tools lecture (20 min)
   ...generate events with Pythia, use Rivet tool to analyze events.

Introduction: Basic resources

We will be using Pythia 8.302 – the latest release. The main Pythia web page is

http://home.thep.lu.se/Pythia/

and you can find the online manual for the 8.3 series here:

http://home.thep.lu.se/~torbjorn/pythia83html/Welcome.html

A copy of the manual is shipped with each Pythia tarball.

Introduction: Resources for this tutorial

More info on today's tutorial can be found at
https://indico.bnl.gov/event/9153/

We will use a `docker` container to make running easy. Installation instructions for `docker` can be found here.

To avoid having to run all docker commands as sudo on Linux systems, create a group "docker" and add yourself:

```
sudo groupadd docker
sudo usermod -aG docker $USER
```

To obtain the `docker` image for this tutorial, simply execute

```
docker pull electronioncollider/pythia-eic-tutorial
```

This will download and store the docker container in `/var/lib/docker`.

Introduction: Getting started with docker container

To get going on this part of the tutorial, we'll start the container with

```
docker run --rm -u `id -u $USER` -v $PWD:$PWD -w $PWD -p 8888:8888 \
   pythia-eic-tutorial
```

...and now we move to the actual tutorial. On to Jupyter!

Introduction: What has happened on Juypter?

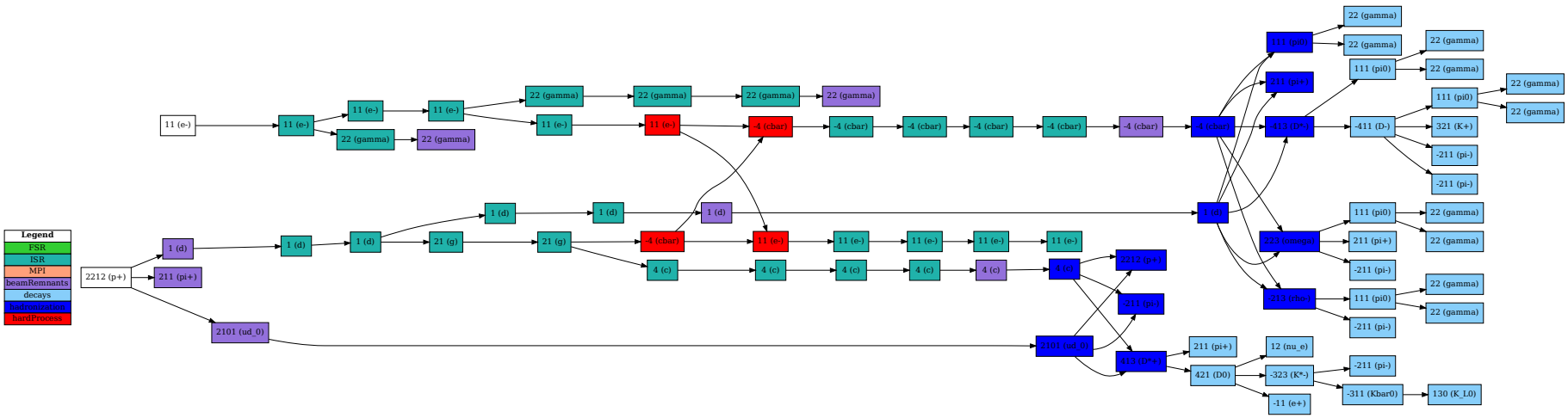
- we've looked at a Pythia event record
  ...in "linear" ASCII format on the terminal
  ...as picture
- we saw the complexity change when switching on/off different components
- we've looked at another type of output: writing to HepMC3 event files.

Group study

I'll now randomly assign you to small groups of people – to work with the notebook yourself!
In the breakout rooms, you can discuss with each other and share results, without disturbing other people.
We will automatically reconvene in 15 minutes.
Try to discuss how the events change if you

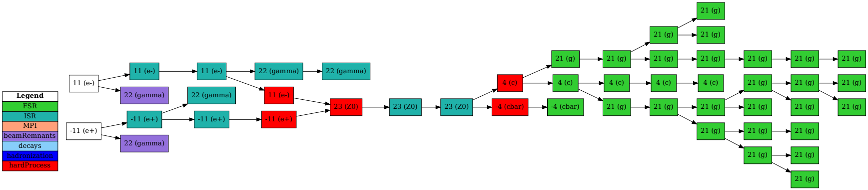
- produce events for different collider setups: ep, e^+e^- and pp
- switch on/off different event generation steps.

Group study: What has happened?



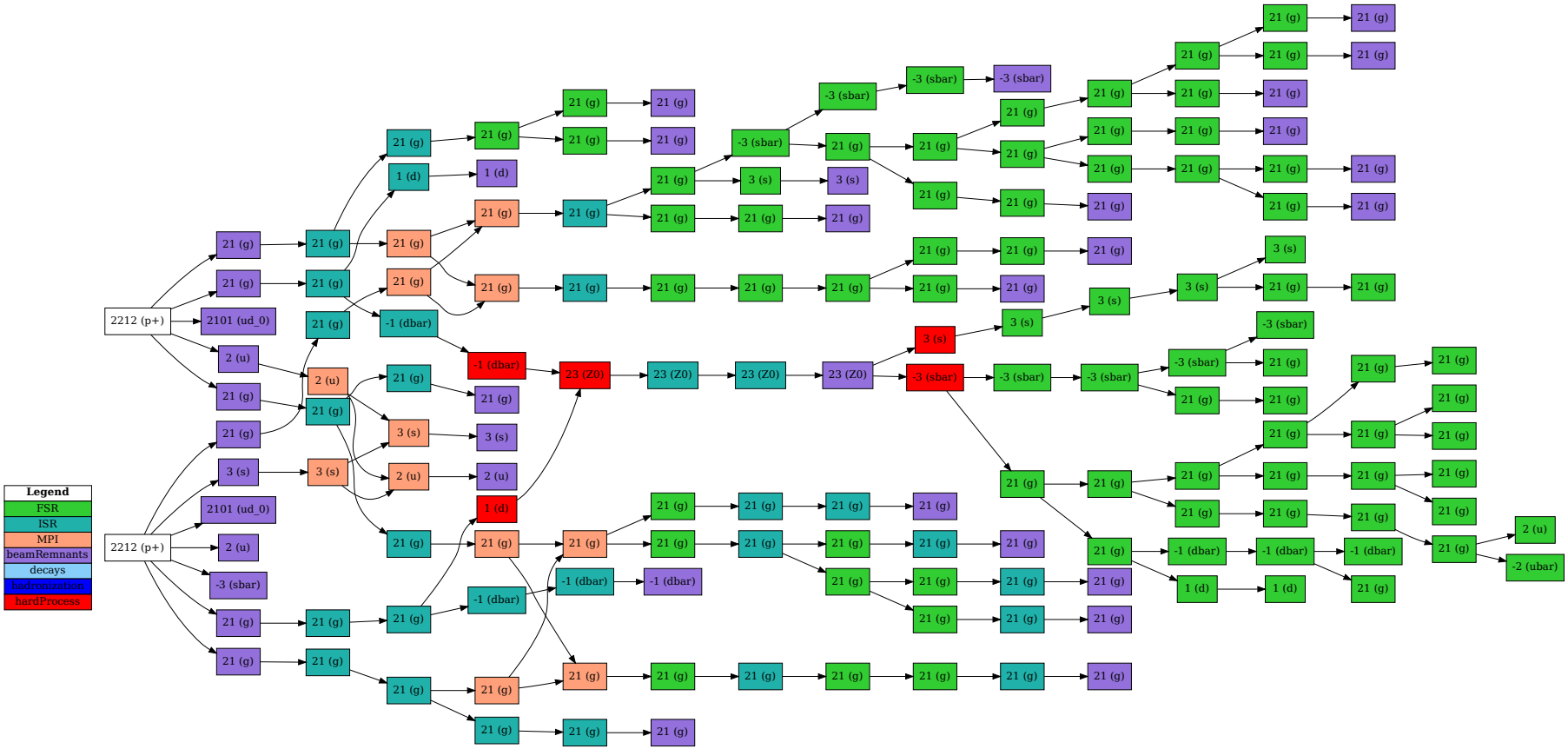
...DIS in full glory

Group study: What has happened?



...no fragmentation at LEP

Group study: What has happened?



...no fragmentation at LHC

Pythia steering crash course: Getting started

Pythia is traditionally steered by

a) using an example main program shipped with the code

b) writing our own steering program

We've already saw an example of a). The approach b) is more powerful – you should at least be familiar with the basics.

Note: If you had to close the terminal to stop the previous `docker run`, you might get a warning `port is already allocated`. You can deallocate the port with

```
sudo docker container ls | tail -n 1 | cut -d " " -f1 | xargs sudo docker rm -f
```

Pythia steering crash course: Getting started

Open a `bash` shell on the `docker` container. Restart the container with

```
docker run --rm -u `id -u $USER` -v $PWD:$PWD -w $PWD -it -p 8888:8888 \
  pythia-eic-tutorial bash
```

then go to

```
cd /usr/local/share/Pythia8/examples
```

Here, you will find all kinds of example programs. Their purpose is documented in the online manual.

In this part, we will now write a small DIS main program that

- generates some DIS events
- histograms the total (vector) transverse momentum of outgoing fundamental particles, so that we can assess the impact of showers and intrinsic parton pT

Pythia steering crash course: A new main program

```
#include "Pythia8/Pythia.h"
using namespace Pythia8;
int main() {
  // Generator.
  Pythia pythia;
  // Beam and process parameters.
  pythia.readString("Beams:idA = -11");
  pythia.readString("Beams:idB = 2212");
  pythia.readString("Beams:frameType = 2");
  pythia.readString("Beams:eA = 27.5");
  pythia.readString("WeakBosonExchange:ff2ff(t:gmZ) = on");
  pythia.readString("SpaceShower:dipoleRecoil = on");
  pythia.readString("Beams:eB = 820.");
  pythia.readString("PhaseSpace:Q2min = 100");
  // Initialize the generator.
  pythia.init();
  // Begin event generation loop.
  int nEvent = 1000;
  for (int iEvent = 0; iEvent < nEvent; ++iEvent) {
    // Generate next event.
    if (!pythia.next()) continue;
    // some analyis goes here.
  }
  // Show statistics.
  pythia.stat();
  return 0;
}
```

Pythia steering crash course: A new main program

We'll use this to understand Pythia. Add a histogram by extending to

```
...
  // Initialize the histograms.
  Hist pt("Transverse hardness", 20, 0.0, 5.0);
  double bw_pt = (5.0-0.0)/20.;
  // Initialize the generator.
  pythia.init();
  // Begin event generation loop.
  int nEvent = 1000;
  for (int iEvent = 0; iEvent < nEvent; ++iEvent) {
    // Generate next event.
    if (!pythia.next()) continue;
    // Event weight, to be filled into the histograms.
    double weight = min(100.,pythia.info.weight());
    // Loop over event record to calculate sum of px and py.
    double px = 0. , py = 0.;
    for (int i = 0; i < pythia.event.size(); ++i){
      if ( pythia.event[i].isFinal()
        && pythia.event[i].idAbs() < 30) { //id<30: all 'fundamental' particles
        px += pythia.event[i].px(); py += pythia.event[i].py();
      }
    }
    // Fill transverse hardness histogram.
    pt.fill(sqrt(px*px + py*py), weight );
  }
...
```

Pythia steering crash course: A new main program

...and then normalize and print to the terminal!

```
...
  // Show statistics.
  pythia.stat();

  // Normalize to cross section [mb].
  double sigmaNorm = pythia.info.sigmaGen() / pythia.info.weightSum();
  pt  *= sigmaNorm / bw_pt;

  cout << pt;
...
```

With this, we can look at the effect of some physics phenomena, e.g.

- Hadronization
  (`HadronLevel:all = on/off`)
- Beam remnants
  (`PartonLevel:Remnants = on/off` and `Check:event = off`)

Pythia steering crash course: A new main program

Before we go on, let's add another simple histogram: The π^+ multiplicity.

```
...
  double area = 0.;
  Hist pth("Hadron pt-spectrum", 40, 0., 1.);
  Hist pt("Transverse hardness", 20, 0.0, 5.0);
...
    // Fill transverse hardness histogram.
    pt.fill(sqrt(px*px + py*py), weight );
    for (int i = 0; i < pythia.event.size(); ++i){
      if ( pythia.event[i].isFinal()
        && pythia.event[i].id() == 211) { //id==211: pi+
        // Charged pion multiplicity as function of pion pT.
        double pT = pythia.event[i].pT();
        pth.fill(pT, weight );
        if (pT >0. && pT < 1.) area += weight;
      }
    }
...
  // Normalize to unit area.
  pth /= area;
  cout << pt << pth;
...
```

Group study

I'll now randomly assign you to small groups of people, as before.
We will automatically reconvene in 20 minutes. Try to discuss

- how the hadron multiplity as function of pT changes when
  - switching off parton showering

```
pythia.readString("PartonLevel:ISR = off");
pythia.readString("PartonLevel:FSR = off");
```

  - changing the intrinsic transverse momentum of partons

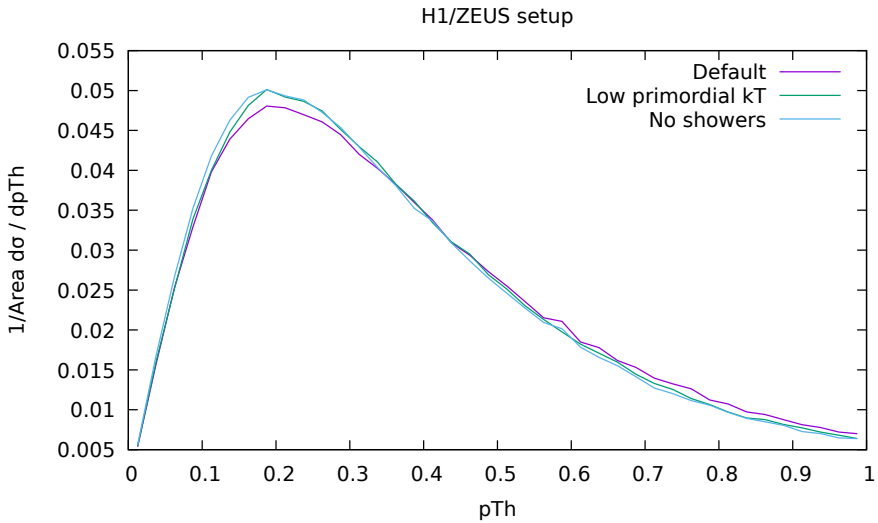
```
pythia.readString("BeamRemnants:primordialKThard = 0.5");
pythia.readString("BeamRemnants:primordialKTsoft = 0.25");
```

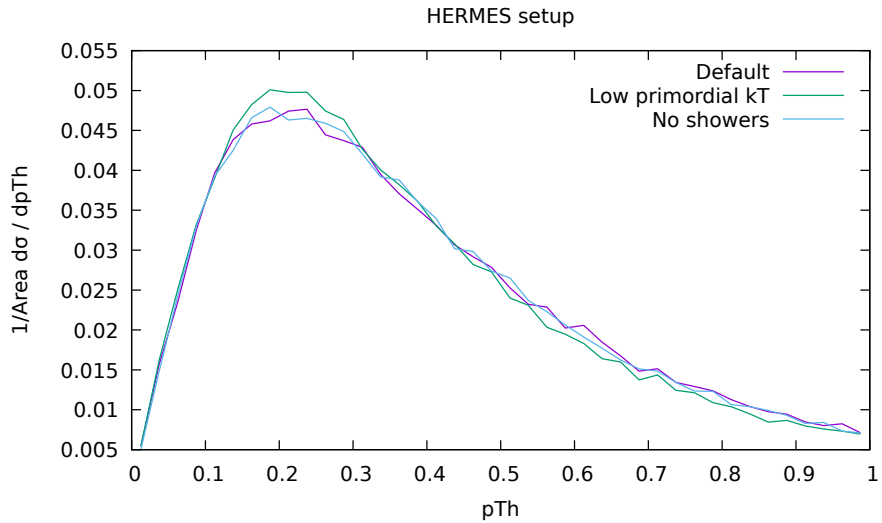
- how does the magnitude of the effects depend on the collider setup?
  Change from a collider to a fixed-target setup

```
pythia.readString("Beams:eB = 0.");
pythia.readString("PhaseSpace:mHatMin=0.1");
pythia.readString("PhaseSpace:Q2min = 2.");
pythia.readString("PhaseSpace:pThatMinDiverge = 0.5");
```

- have a look at the kaon multiplicity, if you have time.

Group study: What has happened?





$\Rightarrow$ At high energies, perturbative corrections are as relevant as intrinsic kT. At lower momentum transfer intrinsic kT dominates.

Using external analysis tools

Now let's go back to a bird's eye view.

In this part, we will generate and store events with a standard main program.

Then, we will analyse the events using the standard analysis provided by the Rivet analysis tool.

On to Jupyter again!

External analysis tools: Note for terminal aficionados

Terminal users: If you want to generate the events from terminal, do

```
docker run --rm -u `id -u $USER` -v $PWD:$PWD -w $PWD -it -p 8888:8888 \
  pythia-eic-tutorial bash
```

and then

```
cd /usr/local/share/Pythia8/examples
make main300
./main300 --setting "WeakBosonExchange:ff2ff(t:gmZ) = on" \
 --setting "Beams:idA = -11" --setting "Beams:idB = 2212" \
 --setting "Beams:frameType = 2" --setting "Beams:eA = 26.7" \
 --setting "Beams:eB = 820" --setting "PhaseSpace:Q2min = 4" \
 --setting "SpaceShower:dipoleRecoil = on" -\
 -setting "phasespace:mhatmin=1.0" --nevents 100000 \
 --hepmc_output dis.hepmc
./main300 --setting "PhotonParton:all = on" \
 --setting "Beams:idA = -11" --setting "Beams:idB = 2212" \
 --setting "Beams:frameType = 2" --setting "Beams:eA = 26.7" \
 --setting "Beams:eB = 820" --setting "PhaseSpace:pTHatMin = 10.0" \
 --setting "PDF:lepton2gamma = on" --setting "phasespace:mhatmin=1.0" --nevents 100000 \
 --hepmc_output photo.hepmc
```

Using external analysis tools: What has happened on Juy

- we've seen how to produce larger HepMC output files with a standard main program
- we've seen how to analyse these files with Rivet
- we've looked at some typical DIS variables, and understood the impact of various Pythia settings
- we saw the different regions of interest for DIS and photoproduction events.
- we ran an archival H1 analysis and compared Pythia to data.

Wrapping it up

This was a two-hour long tutorial for new users of Pythia 8.

I hope you now have

- an understanding of the aim and scope of general-purpose event generators
- some examples and resources on using Pythia from Python/Jupyter
- a starting point on how to traditionally use Pythia from terminal
- and example of using an external event analysis tool

Don't hesitate to contact me (stefan.prestel@thep.lu.se) or Pythia (pythia8@projects.hepforge.org)

I hope you'll use *and help improve* Pythia in the future!