Heavy-ion tutorial(based on Pythia8)

1. Download the tutorial of heavy-ion

```
# using git clone command to download the tutorial
# you can download it in host machine or pythia container

# show my working directory in host machine, you can start wherever you want.
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021$ ls
herwig-tutorial  pythia8-tutorial  sherpa-tutorial

# creat a folder for HI-tutorial
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021$ mkdir HI-tutorial
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021$ ls
herwig-tutorial  HI-tutorial  pythia8-tutorial  sherpa-tutorial

# run images to creat container for HI
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021$ docker run -it --rm
-u $(id -u $USER)  -v $PWD:$PWD -w $PWD hepstore/rivet-pythia
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021$ ls
HI-tutorial  herwig-tutorial  pythia8-tutorial  sherpa-tutorial

# download tutorial using git clone command
# https://gitlab.com/hepcedar/mcnet-schools/beijing-2021
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021$ git clone
https://gitlab.com/hepcedar/mcnet-schools/beijing-2021.git
Cloning into 'beijing-2021'...
remote: Enumerating objects: 200, done.
remote: Counting objects: 100% (200/200), done.
remote: Compressing objects: 100% (106/106), done.
remote: Total 200 (delta 76), reused 170 (delta 57), pack-reused 0
Receiving objects: 100% (200/200), 3.16 MiB | 1.61 MiB/s, done.
Resolving deltas: 100% (76/76), done.

# beijing-2021 folder found in the current directory
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021$ ls
HI-tutorial  beijing-2021  herwig-tutorial  pythia8-tutorial  sherpa-tutorial

# copy the materials of heavy-ion tutorial from beijing-2021 to my working
directory HI-tutorial
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021$ cp -r beijing-
2021/heavy-ions/* HI-tutorial/
```

2. Run code

Here we need to modify the Makefile.inc according to the our env.

```
# enter HI-tutorial and try to run
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ ls
Makefile  Makefile.inc  README.md  slides  test70.cc
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ make
test70
Makefile:46: *** Error: PYTHIA must be built, please run "make" in the top PYTHIA
directory.  Stop.
```

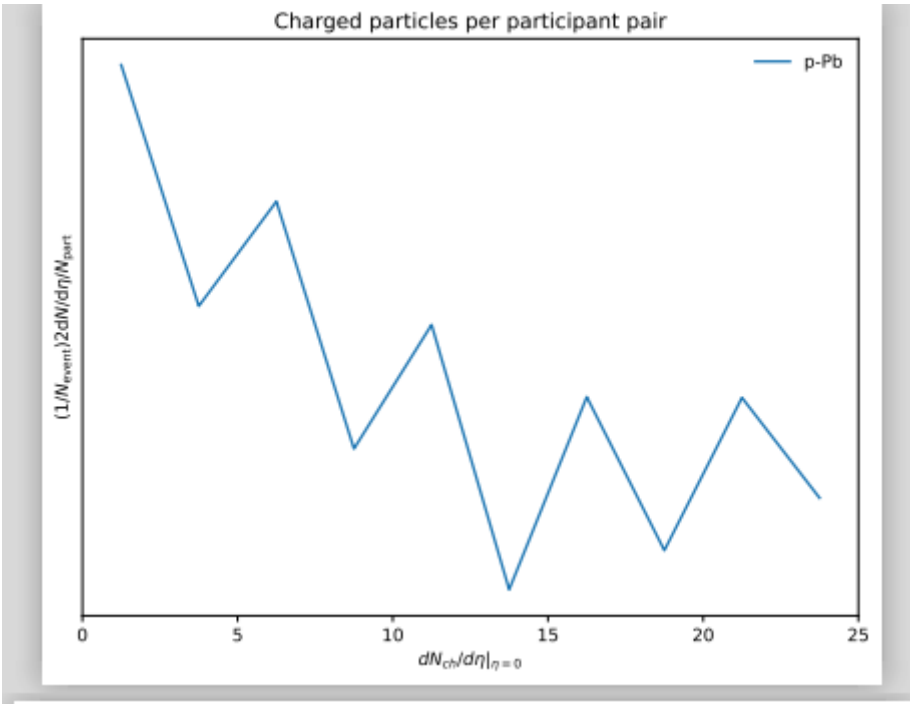
```
# some problem in env setting of Pythia. To solve it
# open the Makefile.inc and replace the following sentences
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ cat
Makefile.inc
# PYTHIA configuration file.
# Generated on Tue 13 Apr 2021 09:08:05 AM CEST with the user supplied options:
# --with-rivet
# --with-hepmc3

# Install directory prefixes.
# These can simply be changed to your own PYTHIA
# installation.
PREFIX_BIN=/home/bierlich/work/pythiaSVN/pythia83/bin              # This is not
our env setting
PREFIX_INCLUDE=/home/bierlich/work/pythiaSVN/pythia83/include
PREFIX_LIB=/home/bierlich/work/pythiaSVN/pythia83/lib
PREFIX_SHARE=/home/bierlich/work/pythiaSVN/pythia83/share/Pythia8
....
********************************************************************************
*******

# modify the directory above
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ cat
Makefile.inc
# PYTHIA configuration file.
# Generated on Tue 13 Apr 2021 09:08:05 AM CEST with the user supplied options:
# --with-rivet
# --with-hepmc3

# Install directory prefixes.
# These can simply be changed to your own PYTHIA
# installation.
# PREFIX_BIN=/home/bierlich/work/pythiaSVN/pythia83/bin
# PREFIX_INCLUDE=/home/bierlich/work/pythiaSVN/pythia83/include
# PREFIX_LIB=/home/bierlich/work/pythiaSVN/pythia83/lib
# PREFIX_SHARE=/home/bierlich/work/pythiaSVN/pythia83/share/Pythia8

PREFIX_BIN=${shell pythia8-config --bindir}
PREFIX_INCLUDE=${shell pythia8-config --includedir}
PREFIX_LIB=${shell pythia8-config --libdir}
PREFIX_SHARE=${shell pythia8-config --datadir}

# you can use pythia8-config --help to check the meaning of pythia8-config
command
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ pythia8-
config --help
Usage: ./pythia8-config [OPTIONS]

Configuration tool for the PYTHIA event generator library. The available
options are defined below. All available options (without arguments) for the
PYTHIA configuration script are also valid for this script. See
"./configuration --help" in the top PYTHIA 8 directory for more details.

Available options.
--help         : Print this help message (also -h, --h, and -help).
--config       : Print the argument(s) used to configure Pythia.
--prefix       : Installation prefix (cf. autoconf). Note that if the
```

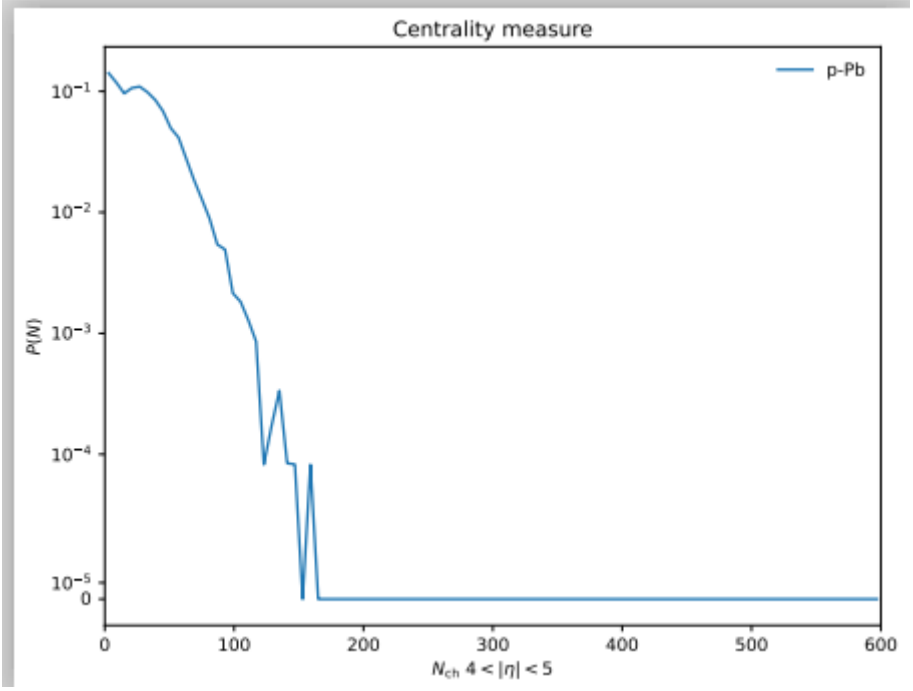
```
                installation is spread over multiple directories, the
                binary directory with the trailing "bin" removed is
                then returned.
--bindir       : PYTHIA binary directory (equivalent to --prefix-bin).
--libdir       : PYTHIA library directory (equivalent to --prefix-lib).
--includedir   : PYTHIA header directory (equivalent to --prefix-include).
--datadir      : PYTHIA share directory (equivalent to --prefix-share).
--xmldoc       : PYTHIA xmldoc directory.
--cxxflags     : Returns the PYTHIA -I flag string needed for compilation.
--cflags       : Equivalent to --cxxflags.
--ldflags      : Returns the PYTHIA -L/-l flag string needed for compilation.
--libs         : Equivalent to --ldflags.
--PACKAGE      : Provides the -I/-L/-l flags needed to link with an external
                PACKAGE from the following list.
--with-PACKAGE : Returns "true" if the package is enabled, otherwise "false".
  evtgen   : Particle decays with the EvtGen decay pacakge, requires HEPMC2.
  fastjet3 : Building of jets using the FastJet package, version 3.
         (run fastjet-config --prefix to see which path to use.)
  hepmc2   : Export PYTHIA events to the HEPMC format, version 2.
  hepmc3   : Export PYTHIA events to the HEPMC format, version 3.
  lhapdf5  : Support the use of external PDF sets via LHAPDF, version 5.
  lhapdf6  : Support the use of external PDF sets via LHAPDF, >= version 6.2.
  powheg   : Hard process production with POWHEGBOX matrix element executables.
  rivet    : Support use of RIVET through direct interface.
  root     : Use ROOT trees and histograms with PYTHIA.
             Note: this option automatically invokes DIR/bin/root-config to set
             the ROOT lib/ and include/ paths. To set your ROOT paths manually
             instead, use --with-root-lib=DIR and --with-root-include=DIR.
  yoda     : Lightweight histogramming package for use with RIVET.
  gzip     : Enable reading of GZIPPED files using the libz library.
  python   : Interface to use PYTHIA in Python.
  mg5mes   : MadGraph matrix element plugins for parton showers.
  openmp   : Multi-threading support via OpenMP.
```

After modification, we can start to run this code

```
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ make
test70
g++ test70.cc -o test70 -I/usr/local/include -O2 -std=c++11 -pedantic -W -Wall -
Wshadow -fPIC  -L/usr/local/lib -Wl,-rpath,/usr/local/lib -lpythia8 -ldl

# generate the executable file test70
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ ls
Makefile  Makefile.inc  README.md  slides  test70  test70.cc

# run test70
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ ./test70

# a lot of warning and erro message. Don't care
# when running is finished
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ ls
Makefile      README.md  test07plot-0.dat  test70     test70plot.py
Makefile.inc  slides     test07plot-1.dat  test70.cc

# using python to draw the plots.  generate a pdf file
```

```
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ python
test70plot.py
Matplotlib created a temporary config/cache directory at /tmp/matplotlib-vmb1tt_9
because the default path (/.config/matplotlib) is not a writable directory; it is
highly recommended to set the MPLCONFIGDIR environment variable to a writable
directory, in particular to speed up the import of Matplotlib and to better
support multiprocessing.
test70plot.py:12: MatplotlibDeprecationWarning: The 'linthreshy' parameter of
__init__() has been renamed 'linthresh' since Matplotlib 3.3; support for the old
name will be dropped two minor releases later.
  plt.yscale('symlog', linthreshy=2.07e-02)
test70plot.py:27: MatplotlibDeprecationWarning: The 'linthreshy' parameter of
__init__() has been renamed 'linthresh' since Matplotlib 3.3; support for the old
name will be dropped two minor releases later.
  plt.yscale('symlog', linthreshy=8.22e-05)

# check the .pdf file
I have no name!@41368e40e573:/home/chenwei/school/MCnet2021/HI-tutorial$ ls
Makefile      README.md  test07plot-0.dat  test07plot.pdf  test70.cc
Makefile.inc  slides     test07plot-1.dat  test70          test70plot.py
```





3. reference

pythia8 manuals: https://pythia.org/manuals/pythia8305/Welcome.html

Heavy ion collision in pythia8: https://pythia.org/manuals/pythia8305/HeavyIons.html

you can learn Angantyr - the default heavy ion model and some parameters setting

4. Exercise 2: Au-Au collisions at RHIC.

set up

```
# First, creat a subfolder for the simulation of Au-Au collisions.
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-tutorial$ ls
Makefile      README.md  test07plot-0.dat  test07plot.pdf  test70.cc
Makefile.inc  slides     test07plot-1.dat  test70          test70plot.py
```

```
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-tutorial$ mkdir
exercise2

(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-tutorial$ ls
exercise2     README.md         test07plot-1.dat  test70.cc
Makefile      slides            test07plot.pdf    test70plot.py
Makefile.inc  test07plot-0.dat  test70

(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-tutorial$ cp
test70.cc Makefile Makefile.inc exercise2/

(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-tutorial$ cd
exercise2/
#  three files in the subfolder exercise2
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-
tutorial/exercise2$ ls
Makefile  Makefile.inc  test70.cc
```

modify the code for Au+Au collisions

change beam type and energy

- PDG id code
- collision CM energy

`mode` **Beams:idA** (`default = 2212`)
The PDG `id` code for the first incoming particle. Allowed codes include
2212 = p, *-2212 = pbar*,
2112 = n, *-2112 = nbar*,
211 = pi^+, *-211 = pi^-*, *111 = pi^0*,
990 = Pomeron (used in diffractive machinery; here mainly for debug purposes),
22 = gamma (for *gamma-gamma* and *gamma-hadron* interactions, more info here),
11 = e^-, *-11 = e^+*,
13 = mu^-, *-13 = mu^+*,
and a few more leptons/neutrinos in a few combinations.
Recently heavy-ion collisions have been implemented in PYTHIA. Therefore a handful of nuclei have been added as allowed incoming beams, using PDG codes of the format `100ZZZAAAI`: 1000020040 = ^{4}He , 1000030060 = ^{6}Li, 1000060120 = ^{12}C, 1000080160 = ^{16}O, 1000290630 = ^{63}Cu, 1000791970 = ^{197}Au, and 1000822080 = ^{208}Pb. More can be added using the function `ParticleData::addParticle`.

`mode` **Beams:idB** (`default = 2212`)
The PDG `id` code for the second incoming particle.

`mode` **Beams:frameType** (`default = 1`; `minimum = 1`; `maximum = 5`)
Choice of frame for the two colliding particles. For options 1 - 3 the beam identities are specified above, while they are obtained by the Les Houches information for options 4 and 5.
`option` **1** : the beams are colliding in their CM frame, and therefore only the CM energy needs to be provided, see `Beams:eCM` below.
`option` **2** : the beams are back-to-back, but with different energies, see `Beams:eA` and `Beams:eB` below. This option could also be used for fixed-target configurations.
`option` **3** : the beams are not back-to-back, and therefore the three-momentum of each incoming particle needs to be specified, see `Beams:pxA` through `Beams:pzB` below.
`option` **4** : the beam and event information is stored in a Les Houches Event File, see `Beams:LHEF` below.
`option` **5** : the beam and event information is obtained by a pointer to an LHAup class instance.

`parm` **Beams:eCM** (`default = 14000.`; `minimum = 0.`)
Collision CM energy, to be set if `Beams:frameType` = 1.

`parm` **Beams:eA** (`default = 7000.`; `minimum = 0.`)
The energy of the first incoming particle, moving in the *+z* direction, to be set if `Beams:frameType` = 2. If the particle energy is smaller than its mass it is assumed to be at rest.

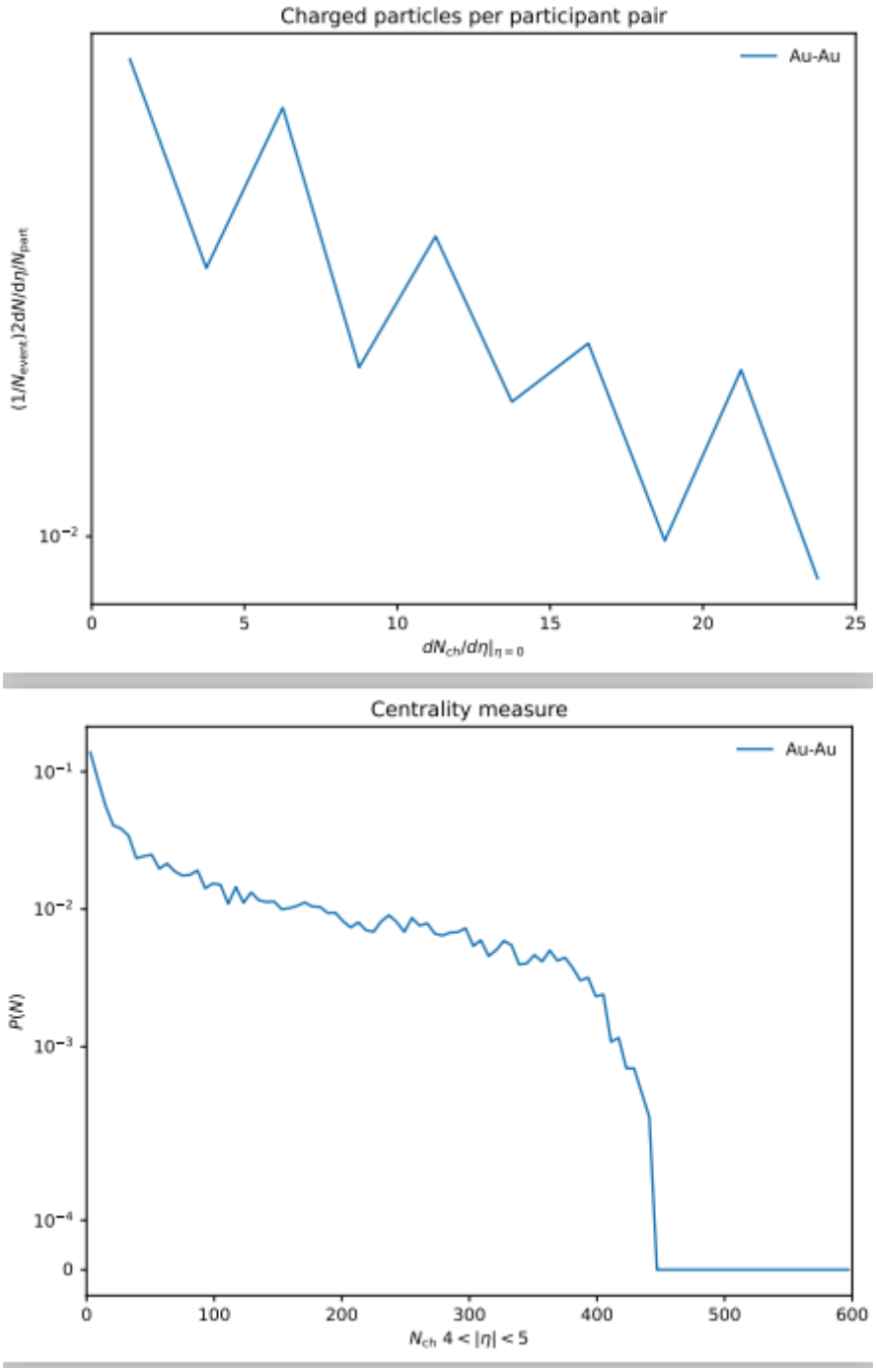
`parm` **Beams:eB** (`default = 7000.`; `minimum = 0.`)
The energy of the second incoming particle, moving in the *-z* direction, to be set if `Beams:frameType` = 2. If the particle energy is smaller than its mass it is assumed to be at rest.

```
in test70.cc, comment the following code lines.

// Beam parameters for p-Pb collisions at LHC, 5TeV.
pythia.readString("Beams:frameType = 2");
pythia.readString("Beams:idA = 2212");
pythia.readString("Beams:idB = 1000822080");
pythia.readString("Beams:eA = 4000.");
pythia.readString("Beams:eB = 1570.");

//replace them with the codes in the following
//Beam parameters for Au+Au collisions at RHIC, 200 GeV
```

```
pythia.readString("Beams:frameType = 1");     //the beams are colliding in their
CM frame
pythia.readString("Beams:idA = 1000791970");
pythia.readString("Beams:idB = 1000791970");
pythia.readString("Beams:eCM = 200.");

//replace 'p-Pb' with 'Au-Au' in the end the file, it will change the label when
drawing the plots
 // Make Python plots.
  hNCh *= 1./sow;
  hNFwd.normalize();
  HistPlot hpl("test70plot"); // <-- this is the name of the generated file.
  hpl.plotFrame("test07plot", hNCh, "Charged particles per participant pair",
    "$dN_{ch}/d\\eta|_{\\eta = 0}$",
    "$(1/N_{\\mathrm{event}}) 2\\mathrm{d}N /
\\mathrm{d}\\eta/N_{\\mathrm{part}}$",
    "-", "p-Pb", true);          // "Au-Au"
  hpl.plotFrame("test07plot", hNFwd, "Centrality measure",
    "$N_{\\mathrm{ch}}~4 < |\\eta| < 5$",
    "$P(N)$",
    "-", "p-Pb", true);         // "Au-Au"
```

Run codes

```
(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-
tutorial/exercise2$ ls
Makefile  Makefile.inc  test70.cc

(base) chenwei@chenwei-ThinkPad-X1-Carbon:~/school/MCnet2021/HI-
tutorial/exercise2$ docker run -it --rm -u $(id -u $USER)  -v $PWD:$PWD -w $PWD
hepstore/rivet-pythia

I have no name!@e50e61e89dc5:/home/chenwei/school/MCnet2021/HI-
tutorial/exercise2$ make test70
g++ test70.cc -o test70 -I/usr/local/include -O2 -std=c++11 -pedantic -W -Wall -
Wshadow -fPIC  -L/usr/local/lib -Wl,-rpath,/usr/local/lib -lpythia8 -ldl

I have no name!@e50e61e89dc5:/home/chenwei/school/MCnet2021/HI-
tutorial/exercise2$ ./test70
# after running, run command:  python test70plot.py to get .pdf file
# check the .pdf file
```



In this part, we also need to change the cross section from 5 TeV to 200 GeV. How to make sure that this is taken care of? Hint: study the output. (listen to the tutor), check the following part in the code.

```
// Initialize the Angantyr model to fit the total and semi-includive
// cross sections in Pythia within some tolerance.
// These parameters work for 5 TeV.
pythia.readString("HeavyIon:SigFitErr = "
                  "0.02,0.02,0.1,0.05,0.05,0.0,0.1,0.0");
// These parameters are suitable for sqrt(S_NN)=5TeV
pythia.readString("HeavyIon:SigFitDefPar = "
                  "17.24,2.15,0.33,0.0,0.0,0.0,0.0,0.0");

// A simple genetic algorithm is run for this many generations
// to fit the parameters. Change if you change COM energy.
pythia.readString("HeavyIon:SigFitNGen = 0");
```

Heavy ion collision in pythia8: https://pythia.org/manuals/pythia8305/HeavyIons.html

Angantyr - the default heavy ion model

The default model in PYTHIA is called Angantyr and is inspired by the old Fritiof model [And86] with improvements described in [Bie16a]. The main idea is to stack parton level events, corresponding to individual nucleon-nucleon sub-collisions, on top of each other and hadronise them together.
Please note: although it is possible to use Rope Hadronisation in heavy ion collisions, these two modules have not yet been validated to work properly together. Also the parameters in the model have not been properly tuned, so the results from running must not be taken as definitive predictions.

To determine which projectile nucleon interacts with which target nucleon, special care is taken to determine in which way the nucleons interact. In a standard *Glauber* calculations one typicaly only cares about if a sub-collision is inelastic or not, but in Angantyr this is divided up, so that each inelastic sub-collision can either be single-diffractive, double-diffractive or absorptive (ie. non-diffractive). To achieve this, Angantyr uses a model with fluctuating radii of the nucleons resulting in a fluctuating nucleon-nucleon cross section inspired by the model by Strikman et al. [Alv13]. The model for this includes a number of parameters which should be fitted to reproduce inclusive nucleon-nucleon cross sections. To be consistent, the values used comes from PYTHIA's internal model of total cross sections.

The default model for nucleon fluctuations has three parameters, the general fitting machinery in `SubCollisionModel` allows for up to eight parameters.

`pvec` **HeavyIon:SigFitDefPar** (`default = {17.24,2.15,0.33,0.0,0.0,0.0,0.0,0.0}`)
These are the default values of the parameters of the `SubCollisionModel` in Angantyr. They will be used as starting point when fitting to the inclusive nucleon cross sections.

The fitting procedure in `SubCollisionModel` is a kind of genetic algorith where a population of parameter values are allowed to evolve for a number of generations. In the end the the parameter set in the final population which gives the best inclusive cross sections is picked. Eight different cross sections may be fitted to but it is possible to select only some of them:

`pvec` **HeavyIon:SigFitErr** (`default = {0.02,0.02,0.1,0.05,0.05,0.0,0.1,0.0}`; `minimum = 0.0`; `maximum = 1.0`)
The relative error assumed in the calculation of goodness of fit corresponding to the different cross sections fitted to. The cross sections are obtained from the `SigmaTotal` and are given as (in order) total, non-diffractive, double diffractive, wounded target, wounded projectile, central diffractive, and elastic cross sections, and in addition the elastic slope parameter. A relative error of zero for one of these cross sections means the corresponding cross section not be used in the fit.

`mode` **HeavyIon:SigFitNInt** (`default = 100000`; `minimum = 0`)
The number of integration points used for each parameter setting to calculate the cross sections.

`mode` **HeavyIon:SigFitNPop** (`default = 20`; `minimum = 1`)
The number individuals (parameter settings) in a population in each generation.

`mode` **HeavyIon:SigFitNGen** (`default = 20`; `minimum = 0`)
The number of generation used in the genetic algorithm. If set to zero, no fitting will be performed and the values in `HeavyIon:SigFitDefPar` will be used.