The JETSCAPE Framework

Hands-on
Part II

Tianyu Dai
Duke University

RHIC & AGS Annual User's meeting
June 8th 2021

Outline

June 8th:

- Overview of JETSCAPE framework
- Run JETSCAPE using container
- Configure JETSCAPE
- Perform parton evolution in static medium
- Implement a new module in JETSCAPE

June 9th:

- JETSCAPE for heavy ion collisions
- Configure JETSCAPE to use realistic hydrodynamic module
- Run p+p collisions
- Extend to A+A collisions and calculate RAA

Questions

Interact on Slack:

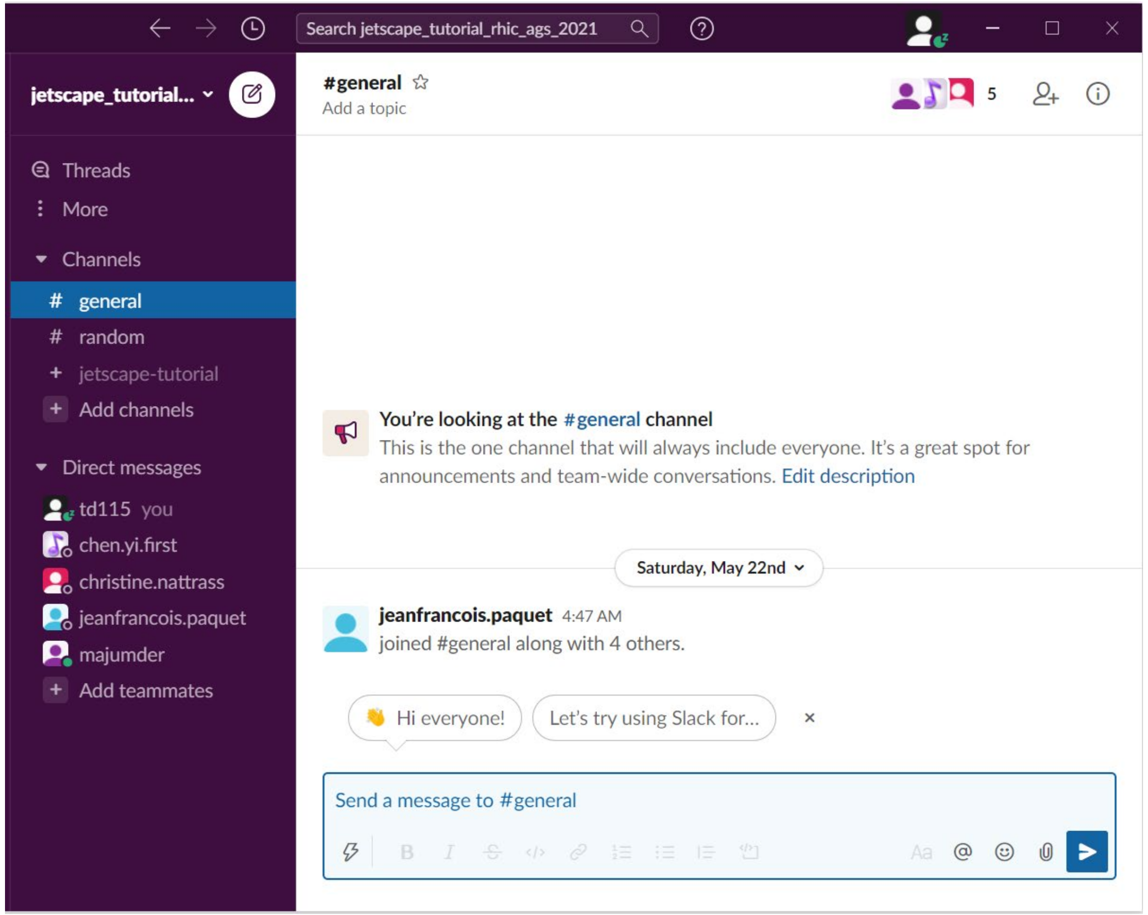
jetscape_tutorial_rhic_ags_2021

- Ask questions : # general
- Questions about installation:

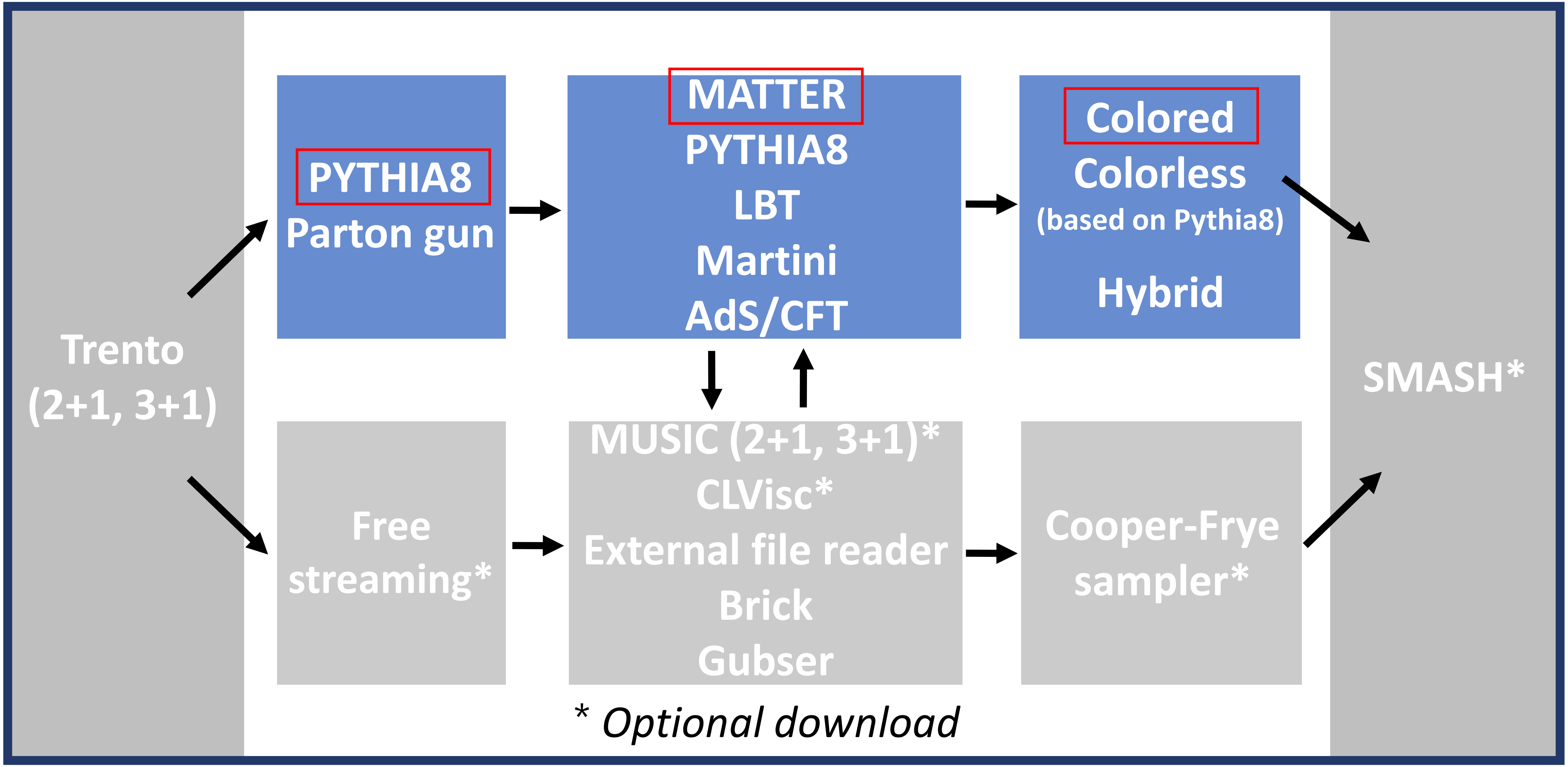
  # installation
- School material: # school_material

Wiki page for tutorial:

https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki



Configuring p+p collisions



Configuring p+p collisions

Modify the configuration file to generate p+p events at

$$\sqrt{S_{NN}} = 2760\ GeV$$

pTHat in Pythia:
transverse momentum in the rest frame of a $2 \rightarrow 2$ processes

Run p+p event in a single pTHatBin

```
./runJetscape /path_to_XML
```

JETSCAPE/config/jetscape_user_pp2760.xml

```
<jetscape>

  <nEvents> 500 </nEvents>

  <outputFilename updated="yes">/home/jetscape-user/JETSCAPE/build/../../JETSCAPE-output/pp2760/500.000000</outputFilename>
  <JetScapeWriterAscii> on </JetScapeWriterAscii>


  <Hard>
    <PythiaGun>
      <pTHatMin updated="yes">500.0</pTHatMin>
      <pTHatMax updated="yes">1380.0</pTHatMax>
      <eCM>2760</eCM>
    </PythiaGun>
  </Hard>


  <Eloss>
    <Matter>
      <Q0> 1.0 </Q0>
      <in_vac> 1 </in_vac>
      <vir_factor> 0.25 </vir_factor>
      <recoil_on> 0 </recoil_on>
      <broadening_on> 0 </broadening_on>
      <brick_med> 0 </brick_med>
    </Matter>
  </Eloss>


  <JetHadronization>
    <name>colored</name>
  </JetHadronization>

</jetscape>
```

→ Set the number of events

→ Change output file name

→ Use Pythia to simulate initial hard scatterings
Set center of mass energy and pTHatBins properly

→ Use vacuum Matter to perform in-vacuum parton evolution

→ Use colored hadronization for p+p collision

Run p+p collisions

```
[Info] 156MB  Run JetScape ...
[Info] 156MB  Number of Events = 500
[Info] 156MB  Run Event # = 0
[Info] 158MB  Run Event # = 100
[Info] 158MB  Run Event # = 200
[Info] 158MB  Run Event # = 300
[Info] 158MB  Run Event # = 400
[Info] 158MB  JetScape finished after 500 events!
[Info] Finished!

CPU time: 12.100152 seconds.
Real time: 12.000000 seconds.
```

Congratulations!

You have run p+p collisions in JETSCAPE successfully!

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Run p+p collisions

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Use a python script to generate and run with configuration files for different pTHatBins

Run the following command:

```
mkdir /rhic-ags-school/JETSCAPE-output/pp2760

cd /rhic-ags-school /JETSCAPE-rhic-ags/analysis

cp ppEventGenerator.py ../build
cd ../build
python3 ppEventGenerator.py
```

analysis/ppEventGenerator.py

```python
#!/usr/bin/env python

import os
import xml.etree.ElementTree as ET
import argparse

pTHat_list = [10., 20., 50., 80., 120., 200., 500., 1380.]
current_path = os.getcwd()

for i, new_pT_hat_min in enumerate(pTHat_list[:-1]):
    with open(current_path+'/../config/jetscape_user_pp2760.xml', 'rb') as xml_file:
        tree = ET.parse(xml_file)
        root = tree.getroot()

        name = root.find('outputFilename')
        file_name = current_path+'/../../JETSCAPE-output/pp2760/%.6f' %(new_pT_hat_min)
        name.text = file_name
        name.set('updated', 'yes')

        hard = root.find('Hard')
        pythia = hard.find('PythiaGun')
        pT_hat_min = pythia.find('pTHatMin')
        pT_hat_max = pythia.find('pTHatMax')

        pT_hat_min.text = str(new_pT_hat_min)
        pT_hat_min.set('updated', 'yes')
        new_pT_hat_max = pTHat_list[i+1]
        pT_hat_max.text = str(new_pT_hat_max)
        pT_hat_max.set('updated', 'yes')

        tree.write(current_path+'/../config/jetscape_user_pp2760.xml', xml_declaration=True, encoding='utf-8')
    os.system(current_path+'/../build/runJetscape '+current_path+'/../config/jetscape_user_pp2760.xml')
```

A list of pTHatBins

Modify XML file

Run JETSCAPE using the modified XML file

Analyze p+p collisions

analysis/ppAnalysis.cc

Check analysis/ppAnalysis.cc :

- Read JETSCAPE Ascii output file
- Extract final hadron information
- Calculate cross section for different p_T

sigmaGen(): the estimated cross section, summed over all allowed process, in unites of mb

```
auto reader=make_shared<JetScapeReaderAscii>(getcwd_string()+"/../../JETSCAPE-output/pp2760/"+
hadron_ct[iBin] = std::vector<int>(pTBin.size()-1, 0);
hadron_ct_sq[iBin] = std::vector<int>(pTBin.size()-1, 0);
std::vector<double> pTSum(pTBin.size()-1, 0.);
while (!reader->Finished())
{
    std::vector<int> hadron_ct_s(pTBin.size()-1, 0);
    reader->Next();

    i_event = reader->GetCurrentEvent();
    auto sigma_gen = reader->GetSigmaGen();
    auto sigma_err = reader->GetSigmaErr();
    if (sigma_err < sigmaErr[iBin])
    {
        sigmaGen[iBin] = sigma_gen;
        sigmaErr[iBin] = sigma_err;
    }

    auto hadrons = reader->GetHadrons();
    auto pdghelper = JetScapeParticleBase::InternalHelperPythia.particleData;
    for (unsigned int iHadron = 0; iHadron < hadrons.size(); iHadron++)
    {
        if (hadrons[iHadron]->pt() < pTBin[0]) continue;
        if (fabs(hadrons[iHadron]->eta()) > 1.) continue;
        auto ID = hadrons[iHadron]->pid();
        auto charge = pdghelper.charge( ID );
        if (charge == 0) continue;
        for (unsigned int ipT = 0; ipT < pTBin.size()-1; ipT++)
            if (hadrons[iHadron]->pt() > pTBin[ipT] && hadrons[iHadron]->pt() <= pTBin[ipT+1])
            {
                hadron_ct_s[ipT]++;
                pTSum[ipT] += hadrons[iHadron]->pt();
                break;
            }
    }
```

Notes on the code:

- Set JETSCAPE Ascii output as reader
- Get cross section for each pTHatBin
- Get final hadrons
- Only record hadrons with $|\eta| < 1$
- Get hadron identity
- Only track charged hadron

Analyze p+p collisions

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Modify JETSCAPE framework to extract cross section information for each pTHatBin

Can be realized by other approaches

JETSCAPE/src/framework/StringTokenizer.cc

```
bool StringTokenizer::isSigmaGenEntry() const {
  if (buffer.length() == 0)
    return false;
  if (buffer.find("# sigmaGen") < 100)
    return true;
  return false;
}

bool StringTokenizer::isSigmaErrEntry() const {
  if (buffer.length() == 0)
    return false;
  if (buffer.find("# sigmaErr") < 100)
    return true;
  return false;
}
```

JETSCAPE/src/reader/JetScapeReader.cc

```
while (getline(inFile, line)) {
  strT.set(line);

  if (strT.isSigmaGenEntry()) {
    string token_s;
    while (!strT.done()) {
      token_s = strT.next();
      if (token_s.compare("#") != 0 && token_s.compare("sigmaGen") != 0) sigmaGen = stod(token_s);
    }
    continue;
  }

  if (strT.isSigmaErrEntry()) {
    string token_s;
    while (!strT.done()) {
      token_s = strT.next();
      if (token_s.compare("#") != 0 && token_s.compare("sigmaErr") != 0) sigmaErr = stod(token_s);
    }
   continue;
  }
```

Modify compiling file

We need to modify the compiling file after adding a new source code, and re-compile the whole package.

Add following lines to JETSCAPE/CMakeList.txt:

```
include_directories(./analysis)
add_executable(ppAnalysis ./analysis/ppAnalysis.cc)
target_link_libraries(ppAnalysis JetScape)
```

Run the following command:

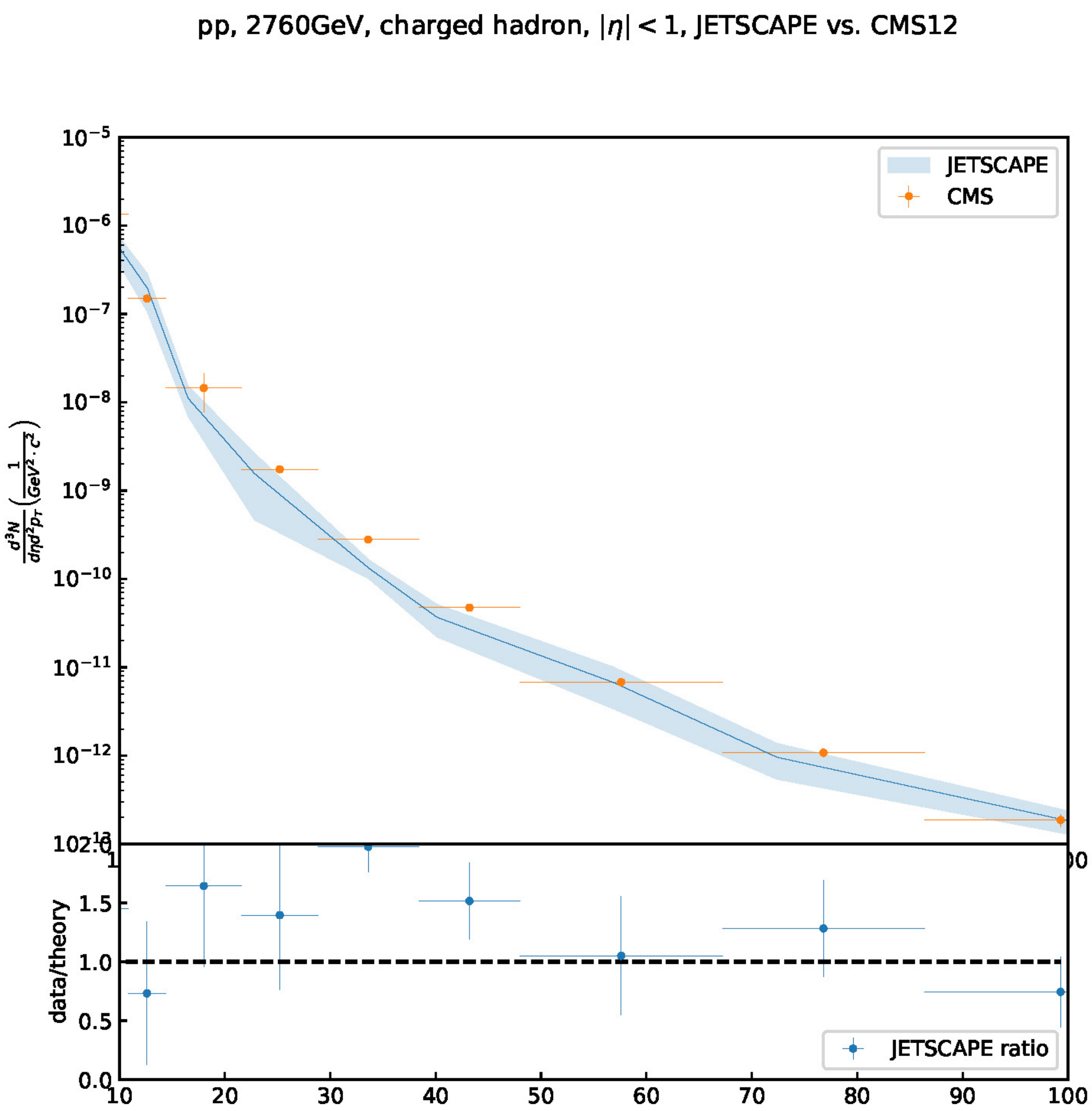
```
cd build
cmake ..
make –j4
./ppAnalysis
```

Tutotial for this practice:
Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

The processed data will be created in JETSCAPE-output/pp2760/pp2760_chargedHadron.txt

p+p results

- Plot p+p cross section using a python script: analysis/plot_pp2760_charged_hadron.py
- Compare it with CMS12 data



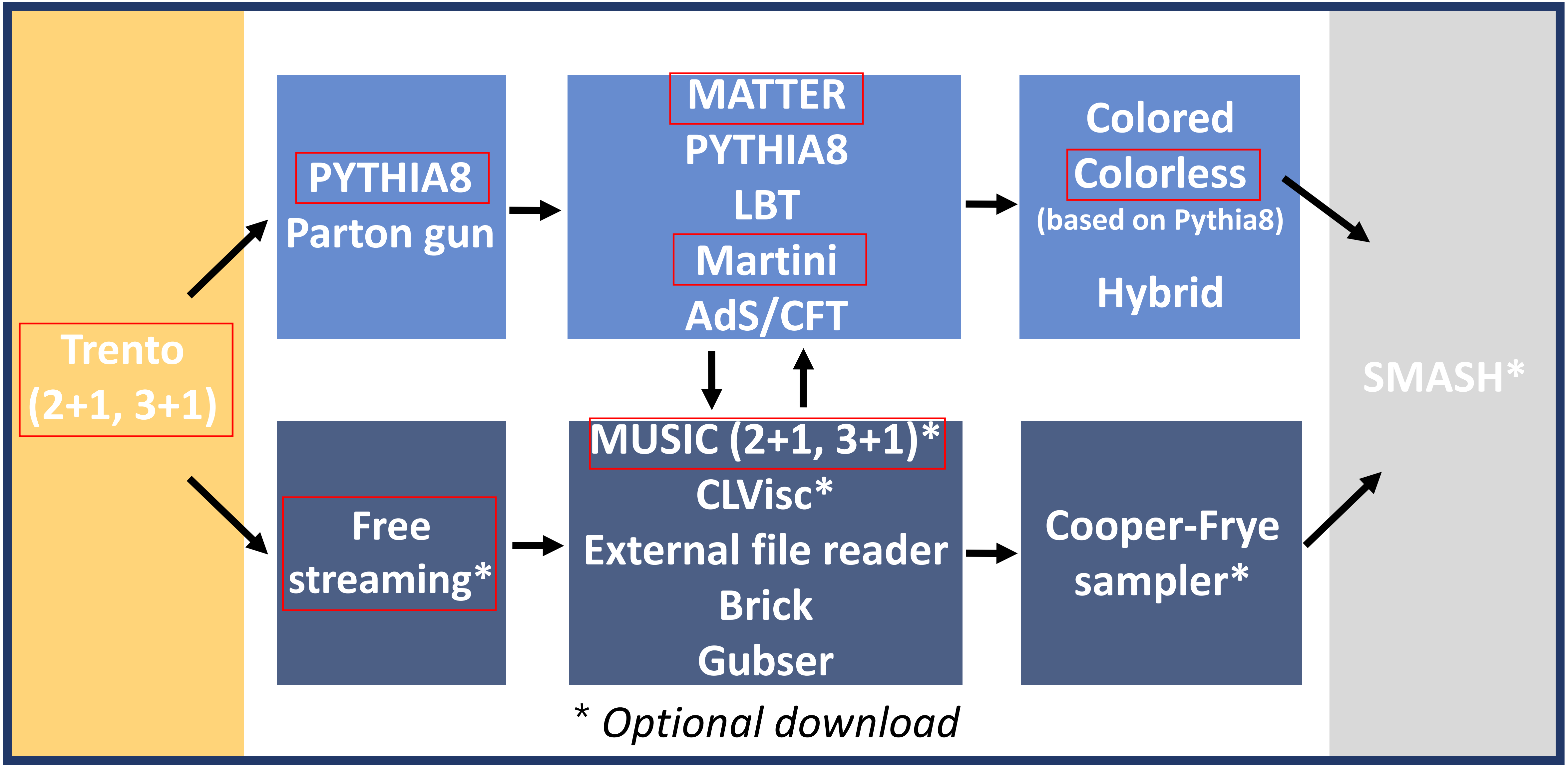
This is only a toy example with bad statistics.

See p+p results obtained using JETSCAPE:

p+p results

JETSCAPE framework: p+p results
A. Kumar et al. (The JETSCAPE collaboration)
Phys. Rev. C **102**, 054906 – Published 10 November 2020

Configuring p+p collisions



Configuring A+A collisions

Modify the configuration file to generate A+A events
Au+Au, $\sqrt{S_{NN}} = 200\ GeV$

JETSCAPE/config/jetscape_user_AA200.xml

Hydro reuse

Initial condition

```
<ReuseHydro> True </ReuseHydro>
<nReuseHydro> 100000 </nReuseHydro>
<Random>
  <seed>0</seed>
</Random>

<IS>

  <grid_max_x> 15 </grid_max_x>
  <grid_max_y> 15 </grid_max_y>
  <grid_max_z> 0.0 </grid_max_z>
  <grid_step_x> 0.3 </grid_step_x>
  <grid_step_y> 0.3 </grid_step_y>
  <grid_step_z> 0.3 </grid_step_z>

  <Trento>
            <PhysicsInputs cross-section="4.2" normalization="13." projectile="Au" sqrts="200" target="Au">
            </PhysicsInputs>
            <CutInputs centrality-high="20" centrality-low="10">
            </CutInputs>

            <TransInputs fluctuation="0.9" nucleon-min-dist="0.4" nucleon-width="0.8" reduced-thickness="0.1">
            </TransInputs>
            <LongiInputs jacobian="0.8" mean-coeff="0.0" skew-coeff="0.0" skew-type="1" std-coeff="100.0">
            </LongiInputs>

  </Trento>
</IS>
```

Pre-equilibrium

Hydro file

```
<Preequilibrium>
  <NullPreDynamics> </NullPreDynamics>
</Preequilibrium>

<Hydro>
  <MUSIC>
    <name>MUSIC</name>
    <MUSIC_input_file>music_input</MUSIC_input_file>
    <output_evolution_to_file>1</output_evolution_to_file>
    <shear_viscosity_eta_over_s>0.08</shear_viscosity_eta_over_s>
    <freezeout_temperature>-1</freezeout_temperature>
    <Perform_CooperFrye_Feezeout>0</Perform_CooperFrye_Feezeout>
  </MUSIC>
</Hydro>
```

High-virtuality energy loss model

Low-virtuality energy loss model

Colorless hadronization

```
<Eloss>

  <deltaT>0.01</deltaT>
  <maxT>20.</maxT>

  <Matter>
    <in_vac> 1 </in_vac>
    <Q0> 1.0 </Q0>
  </Matter>

  <Martini>
    <alpha_s> 0.3 </alpha_s>
    <pcut> 2. </pcut>
    <Q0> 1.0 </Q0>
    <hydro_Tc> 0.16 </hydro_Tc>
  </Martini>

</Eloss>


<JetHadronization>
  <name>colorless</name>
</JetHadronization>
```

Run A+A collisions

Run A+A events in one pTHatBin: [60, 100] (GeV)

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Use the following command:

```
mkdir /rhic-ags-school/JETSCAPE-output/AuAu200
cd /rhic-ags-school/JETSCAPE-rhic-ags/build
./runJetscape ../config/jetscape_user_AA200.xml
```

For a full simulation, we should run A+A events with different pTHatBins, as what we did for p+p collisions.

To save time, we only run in one pTHatBin in today's practive.

Try to use analysis/AAEventgenerator.py to run A+A events at various pTHatBins when you have time:

```
cd /rhic-ags-school/JETSCAPE-rhic-ags/build
cp ../analysis/AAEventgenerator.py .
python3 AAEventgenerator.py
```

Run A+A collisions

```
[Info] 166MB   Load TRENTo density and ncoll density to JETSCAPE memory
[Info] 166MB  10000 density elements
[Info] 166MB  10000 ncoll elements
[Info] 166MB   TRENTO event generated and loaded
[Info] 167MB  Initialize density profiles in MUSIC ...
 171.4 MB OpenMP: using 4 threads.
 171.4 MB initArena
 171.4 MB Using Initial_profile=42. Overwriting lattice dimensions:
 171.4 MB neta = 1, nx = 100, ny = 100
 171.4 MB deta=0.1, dx=0.3, dy=0.3
 171.4 MB x_size = 30, y_size = 30, eta_size = 0
 176.3 MB Grid allocated.
 176.3 MB  ----- information on initial distribution -----
 176.3 MB initialized with a JETSCAPE initial condition.
 176.3 MB initial distribution done.
[Info] 176MB  initial density profile dx = 0.3 fm
[Info] 176MB  number of source terms: 0, total E = 0 GeV.
[Info] 176MB  running MUSIC ...
 176.3 MB Freeze out at a constant temperature T = 0.137 GeV, e_fo = 0.120635 GeV/fm^3
 176.3 MB eps_max = 31.9008 GeV/fm^3, rhob_max = 0 1/fm^3, T_max = 0.372326 GeV.
 176.3 MB  Done time step 0/1500 tau = 0.6 fm/c
 176.3 MB eps_max = 30.5827 GeV/fm^3, rhob_max = 0 1/fm^3, T_max = 0.36867 GeV.
 176.3 MB  Done time step 1/1500 tau = 0.62 fm/c
 176.3 MB eps_max = 29.3641 GeV/fm^3, rhob_max = 0 1/fm^3, T_max = 0.365186 GeV.
 176.3 MB  Done time step 2/1500 tau = 0.64 fm/c
 176.3 MB eps_max = 28.2332 GeV/fm^3, rhob_max = 0 1/fm^3, T_max = 0.361856 GeV.
 176.3 MB  Done time step 3/1500 tau = 0.66 fm/c
```

Generate hydrodynamic events successfully!

```
[Info] 156MB  Run JetScape ...
[Info] 156MB  Number of Events = 500
[Info] 156MB  Run Event # = 0
[Info] 158MB  Run Event # = 100
[Info] 158MB  Run Event # = 200
[Info] 158MB  Run Event # = 300
[Info] 158MB  Run Event # = 400
[Info] 158MB  JetScape finished after 500 events!
[Info] Finished!

CPU time: 12.100152 seconds.
Real time: 12.000000 seconds.
```

Run A+A collisions successfully!

Run p+p collisions

Write a new configuration file config/jetscape_user_pp200.xml based on config/jetscape_user_pp2760.xml to run p+p collisions at $\sqrt{S_{NN}} = 200$ GeV

```
<nEvents> 2500 </nEvents>

<outputFilename> /../../JETSCAPE-output/pp200/60.000000 </outputFilename>
<JetScapeWriterAscii> on </JetScapeWriterAscii>


<Hard>
  <PythiaGun>
    <pTHatMin>60.0</pTHatMin>
    <pTHatMax>100.0</pTHatMax>
    <eCM>200</eCM>
  </PythiaGun>
</Hard>
```

Run p+p events in one pTHatBin:

```
./runJetscape ../config/jetscape_user_pp200.xml
```

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Calculate RAA

Data for A+A events: JETSCAPE-output/AuAu200/60.000000.dat
Data for p+p events: JETSCAPE-output/pp200/60.000000.dat

Use analysis/AAAnalysis.cc to extract **final state charged hadron** from raw data

To compile analysis/AAAnalysis.cc with JETSCAPE, add the following line to JETSCAPE-rhic-ags/CMakeList.txt:

```
add_executable( AAAnalysis ./analysis/AAAnalysis.cc )
target_link_libraries( AAAnalysis JetScape )
```

Re-compile and build JETSCAPE:

```
cd rhic-ags-school/JETSCAPE-rhic-ags/build
cmake –DUSE_MUSIC=ON ..
make -j
```

Tutotial for this practice: Find in Github wiki page!
https://github.com/TianyuDai/JETSCAPE-rhic-ags/wiki/Hadron-production-in-proton-proton-with-the-JETSCAPE-framework

Calculate RAA

use the python script analysis/plot_RAA.py to generate a figure of charge hadron R_{AA} at $\sqrt{S_{NN}} = 200$ GeV

Run AAAnalysis to extract information for both p+p and A+A:

```
cd /rhic-ags-school/JETSCAPE-ags/rhic/analysis
./AAAnalysis ../../JETSCAPE-output/AuAu200 ../../JETSCAPE-output/AuAu200/AuAu200_chargedHadron.txt
./AAAnalysis ../../JETSCAPE-output/pp200 ../../JETSCAPE-output/AuAu200/pp200_chargedHadron.txt
```

analysis/AAAnalysis.cc has two input parameters:

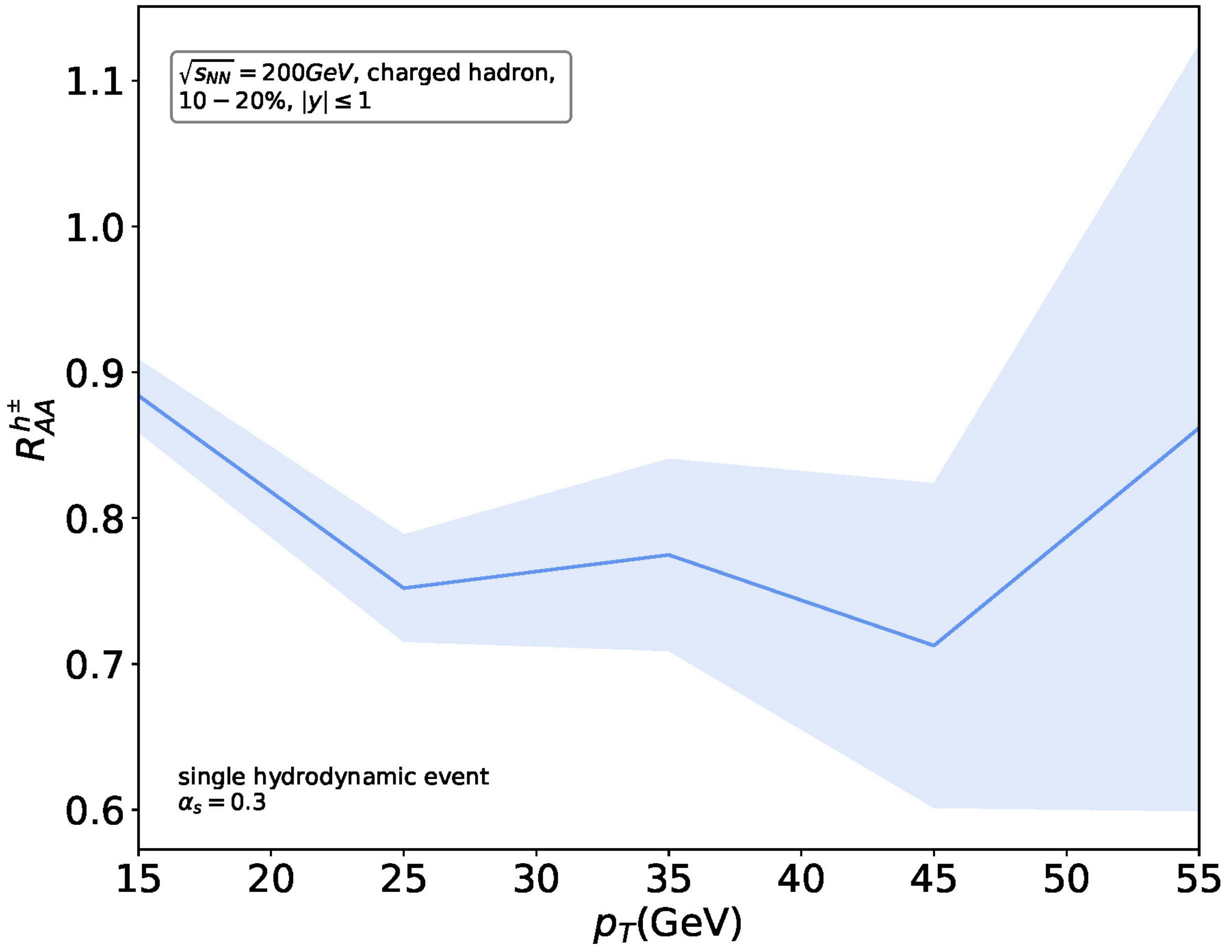
- *brick_out.dat is the filename of the Ascii output file*
- *my_final_state_partons.txt is the file to save the final state parton information*

Run plot_RAA.py to generate a figure of R_{AA}

```
cd rhic-ags-school/JETSCAPE-rhic-ags/analysis
python3 plot_RAA.py
```

Calculate RAA

use the python script analysis/plot_RAA.py to generate a figure of charge hadron R_{AA} at $\sqrt{S_{NN}} = 200$ GeV



Large error band for limited time:

- Small number of events
- Single pTHatBin
- Coarse hydro grid
- Single hydro event
- ……

Results can be different given the random seed