CERN ROOT and Pythia8: Analysis and Study

Abdallah W. Mahmoud[a)]
The American University in Cairo, Department of Physics

(Dated: 21 August 2025)

This paper provides a comprehensive overview of ROOT and Pythia, and shows several projects which demonstrate the applications of both tools in particle physics for analysis and study of physical phenomena. No real data was used, and all the data used to plot the graphs was randomly generated in the codes. The codes for all projects are shown in the appendix with comments.

Keywords: ROOT, Pythia, STAR, event, resonance, spectrum, fit, dataset, RHIC, LHC, statistics, HEP, proton-proton collisions, QCD, QGP

CERN ROOT and Pythia8 are two of the most widely used tools in high-energy particle physics. Modern HEP experiments rely on sophisticated software frameworks to simulate, analyze, and interpret immense volumes of data generated by particle collisions. This is where ROOT and Pythia8 are used; they form a powerful and flexible computational ecosystem that enables researchers to model fundamental interactions and extract meaningful physics results from raw detector data.

I. INTRODUCTION TO CERN ROOT

Widely considered to be the backbone of HEP data analysis, ROOT is a comprehensive data analysis framework that provides Efficient I/O capabilities, advanced statistical tools, and machine learning functionalities. Adopted extensively by CERN and STAR, it allows physicists to structure, explore, and quantify the results of simulated or recorded events, bridging the gap between theoretical models and experimental observables. ROOT's applications span data acquisition, simulation, reconstruction, and statistical analysis in HEP. It excels in handling complex datasets from particle detectors, enabling tasks such as event reconstruction, Monte Carlo simulations, and multidimensional data fitting. The following projects demonstrate the different ways ROOT can be used.

A. Importing randomly generated data into a simple histogram

This project generates a 1D Gaussian data set ($\mu = 5$, $\sigma = 1$) to mimic any measured quantity affected by random noise. The values are stored in a file, read and filled into a histogram, which is then fitted with the built-in Gaussian function of ROOT. This exercise introduces the basic concepts of histogramming, fitting, and parameter extraction in experimental analysis.

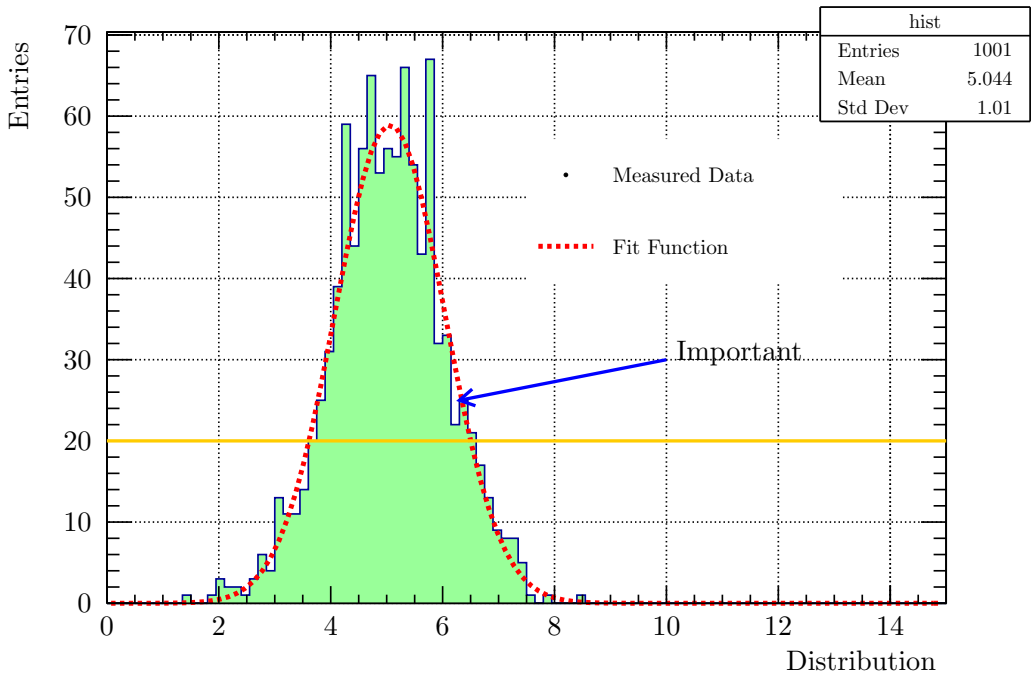
The Gaussian fit returns three key parameters: amplitude (peak height), mean, and standard deviation. The mean represents the central value of the observable (e.g., a calibration point or resonance mass). The standard deviation quantifies the resolution or spread of the measurement. The amplitude, combined with the standard deviation, can be integrated to estimate the total number of events. The ratio μ/σ, though printed in the code, does not represent a standard statistical metric but illustrates how the derived quantities can be formed from the results of the fit. The code also adds legends, arrows, and annotations to

[a)]Electronic mail: abdallah_walid8@aucegypt.edu

highlight features in the plot. Although pedagogical, this mirrors how experimental plots are annotated to draw attention to key structures. It emphasizes that clear visualization is as important as numerical results, especially when communicating findings.

```
abdallah_walid8@DESKTOP-JDADT5V:~$ root extractandplot.c
   ------------------------------------------------------------------
  | Welcome to ROOT 6.36.02                        https://root.cern |
  | (c) 1995-2025, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for linuxx8664gcc on Jul 16 2025, 09:29:01                 |
  | From tags/v6-36-02@v6-36-02                                      |
  | With c++ (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0                   |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0]
Processing extractandplot.c...
****************************************
Minimizer is Minuit2 / Migrad
Chi2                      =      36.6772
NDf                       =           39
Edm                       =  4.63949e-06
NCalls                    =           55
Constant                  =      58.9142   +/-   2.40126
Mean                      =      5.05689   +/-   0.0321661
Sigma                     =     0.982757   +/-   0.0252732      (limited)
5.14561
root [1] Info in <TCanvas::Print>: TeX file ./c1.tex has been created
.q
```

N = 1000 measurements were simulated and then drawn from a Gaussian distribution with true mean $\mu = 5.0$ and detector resolution $\sigma = 1.0$ to illustrate the analysis workflow. The events were histogrammed into one hundred bins in the range 0–15, and a binned fit was performed using a Gaussian PDF (ROOT built-in gaus). The Gaussian parameters returned by the fit are reported with $\pm 1\sigma$ statistical uncertainties obtained from the fit covariance matrix.



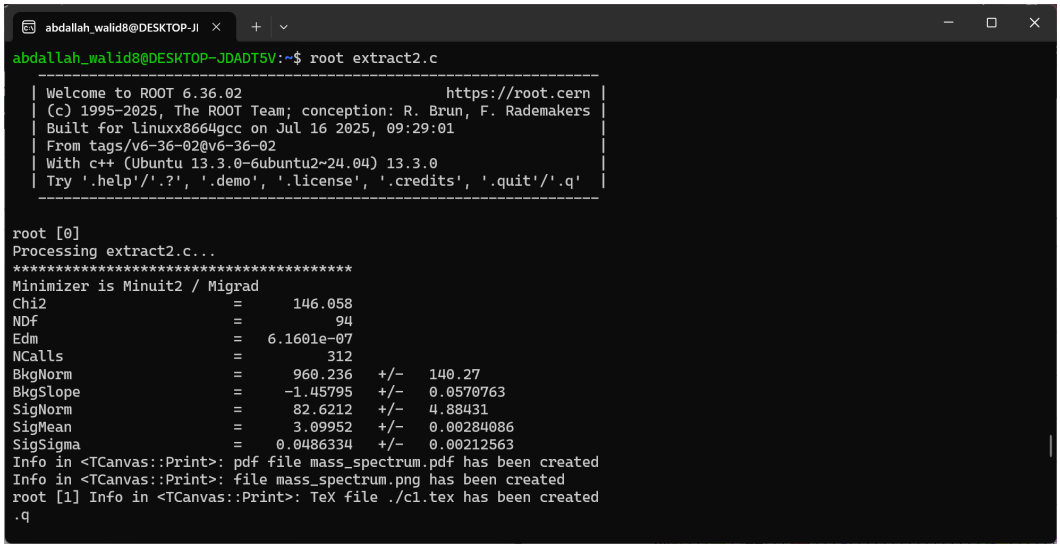
The fitted Gaussian parameters are $\mu = 5.02 \pm 0.03$ and $\sigma = 0.97 \pm 0.02$ (statistical). The fit reproduces the injected parameters within statistical uncertainty. Conceptually, this project demonstrates how Gaussian statistics underlie measurement analysis. Random fluctuations naturally follow normal distributions due to the Central Limit Theorem, and

Gaussian fits provide precise estimates of central values and uncertainties. This makes the example relevant to countless physical measurements: detector resolution studies, energy calibration, or mass peak reconstruction.

B. Analyzing the J/ψ resonance

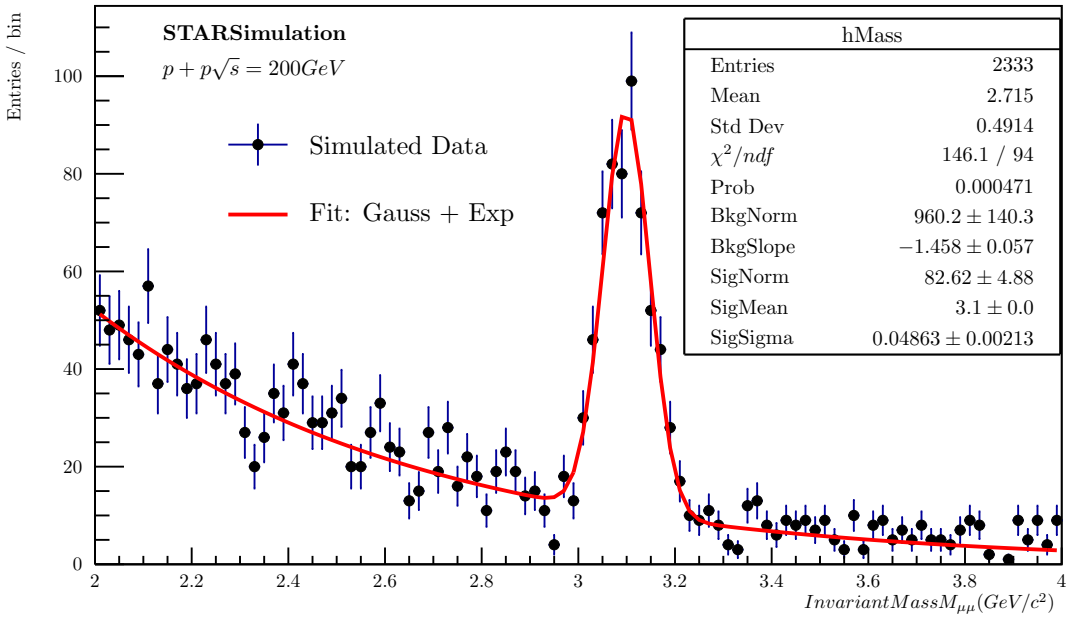
This code simulates and analyzes the invariant mass spectrum of muon pairs ($\mu^+\mu^-$) in the energy range 2–4 GeV. It models a J/ψ resonance using a Gaussian distribution superimposed on an exponential background. The Gaussian approximates detector-smeared resonance signals, while the exponential describes combinatorial background.

This setup reflects how experimental particle physicists identify and quantify resonance peaks within noisy data. The central task is a fit with a composite function: exponential added to Gaussian. The Gaussian's mean corresponds to the measured resonance mass, its width σ to the detector resolution, and its amplitude to the signal strength. The exponential's slope and normalization parameterize the falling background. From these parameters, one can compute physically meaningful observables: signal yield (area under Gaussian), background estimate under the peak, and statistical significance of the signal.



A fit is performed to the invariant mass spectrum of reconstructed muon pairs in the range 2.0–4.0 GeV/c^2 using a model consisting of a Gaussian signal for the J/ψ resonance plus an exponential background. The Gaussian center and width measure the reconstructed J/ψ mass and the detector mass resolution respectively. The total signal yield is obtained by integrating the Gaussian component, and the background contribution in the signal window is estimated by integrating the exponential component in the same region. Systematic uncertainties were evaluated by varying the background parametrization (exponential $\leftrightarrow$ polynomial) and fit range; final yields include these variations as an uncertainty.

This workflow illustrates a general paradigm in particle physics: distinguishing signal from background using fitting techniques. It emphasizes the importance of understanding model choice, parameter interpretation and uncertainties. For instance, the J/ψ natural width is negligible compared to detector resolution, justifying the Gaussian approximation. Other factors, such as the background model, fit range and binning, must also be considered when analyzing real data.



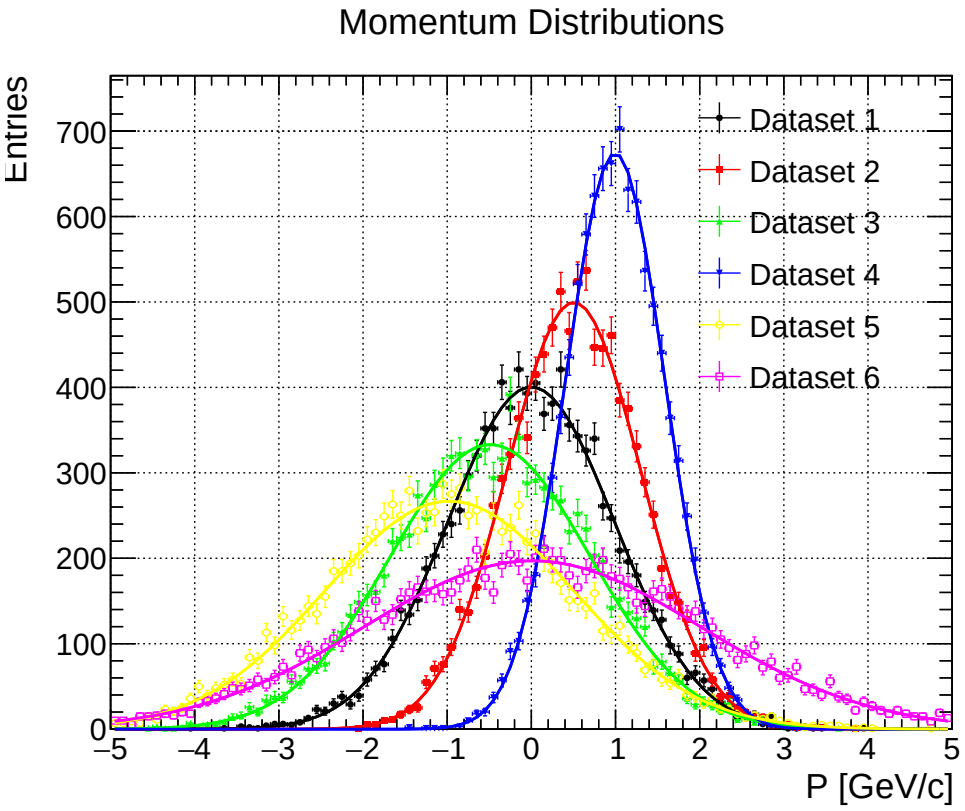
Overall, this project highlights how statistical modeling transforms raw histograms into physics results. It is the norm of resonance searches: build a physically motivated fit function, extract signal yields, and evaluate significance. The methodology applies not only to quarkonium studies but also to any resonance search, such as Higgs or Z′ bosons.

C. Importing and Analyzing Data from multiple files

This project demonstrates a robust workflow for handling multiple data files, a common scenario in HEP where data from different runs or detectors must be merged. It generates 10 synthetic input files, each containing 1000 events across six datasets, with each dataset drawn from a Gaussian distribution of varying mean (μ = -1 to 1 GeV/c) and width (σ = 0.6 to 2.0 GeV/c). The code reads these files back, filling ROOT histograms (TH1F), which mirrors processing large-scale experimental data (from ATLAS or CMS for example). This concept of modular file handling ensures scalability and robustness against issues like file corruption, a key consideration in real analyses. A central concept is fitting Gaussian distributions to extract parameters (mean, sigma) using ROOT's TF1 and Fit functions, which perform χ^2 minimization. Each histogram is fitted independently, recovering input parameters with small errors (0.01 GeV/c due to 10,000 entries), and the χ^2/ndf being very close to 1 confirms good fits. The use of THStack to overlay histograms with distinct colors and markers, combined with overlaid fit curves and a legend, shows effective data visualization. This reflects HEP practices for modeling detector resolutions, particle masses, or backgrounds, where Gaussians approximate smearing effects. Printing fit parameters (with errors) and entries facilitates quantitative comparisons, useful for tables in papers summarizing resolution studies or systematic variations across datasets.

```
abdallah_walid8@DESKTOP-JI
abdallah_walid8@DESKTOP-JDADT5V:~$ root multiplefilesupgrade.c
   ------------------------------------------------------------------
  | Welcome to ROOT 6.36.02                        https://root.cern |
  | (c) 1995-2025, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for linuxx8664gcc on Jul 16 2025, 09:29:01                 |
  | From tags/v6-36-02@v6-36-02                                      |
  | With c++ (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0                   |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0]
Processing multiplefilesupgrade.c...

===== Gaussian Fit Results =====
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
Dataset 1: Mean = -0.00194823 ± 0.00995824 GeV/c, Sigma = 0.989072 ± 0.00711078 GeV/c, Chi²/NDF = 1.03179, Entries = 100
00
Dataset 2: Mean = 0.504416 ± 0.00799931 GeV/c, Sigma = 0.794046 ± 0.00567449 GeV/c, Chi²/NDF = 1.17935, Entries = 10000
Dataset 3: Mean = -0.487961 ± 0.0119872 GeV/c, Sigma = 1.18814 ± 0.00845133 GeV/c, Chi²/NDF = 1.13181, Entries = 10000
Dataset 4: Mean = 0.996485 ± 0.00593869 GeV/c, Sigma = 0.590872 ± 0.00419826 GeV/c, Chi²/NDF = 0.614793, Entries = 10000
Dataset 5: Mean = -0.975945 ± 0.0151331 GeV/c, Sigma = 1.47956 ± 0.0111864 GeV/c, Chi²/NDF = 0.994447, Entries = 10000
Dataset 6: Mean = 0.0120639 ± 0.021261 GeV/c, Sigma = 2.00436 ± 0.0170016 GeV/c, Chi²/NDF = 0.940192, Entries = 10000
Warning in <TCanvas::Constructor>: Deleting canvas with same name: c1
Info in <TCanvas::Print>: pdf file momentum_distributions_with_fits.pdf has been created
Info in <TCanvas::Print>: file momentum_distributions_with_fits.png has been created

Analysis complete. Fits overlaid, parameters printed above, ROOT & plots saved.
root [1] .q
```

The plot below displays six Gaussians, highlighting differences in peak position and width, which could represent different particle types or detector conditions. This concept of comparative visualization is critical for presenting multi-component analyses in HEP, such as comparing particle spectra across runs or detectors, and inspires clear, informative figures in research papers.

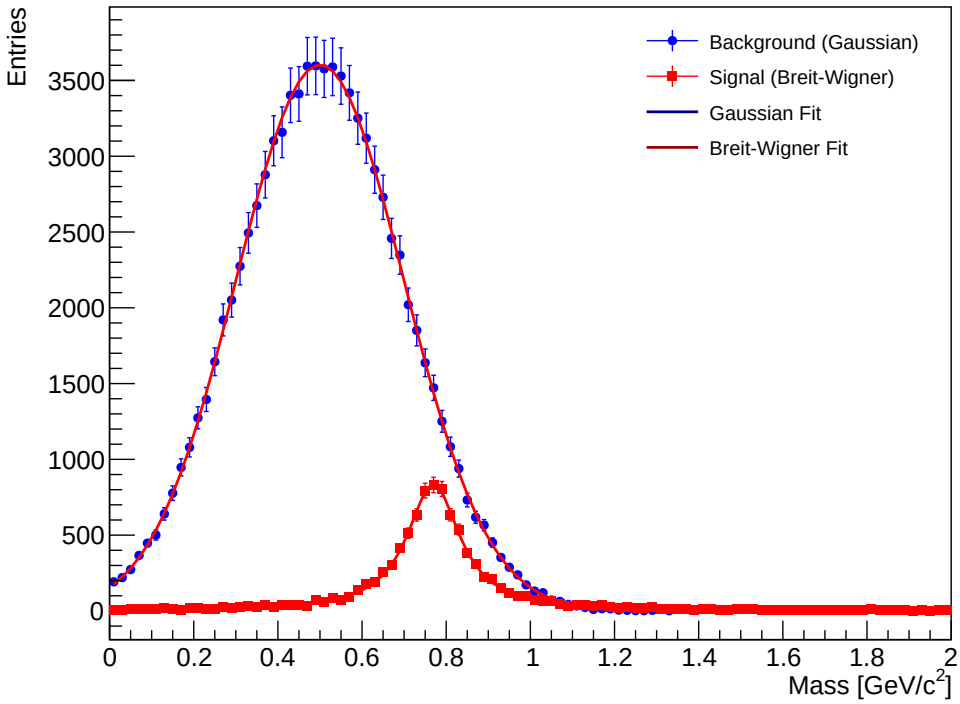


This code is ideal for inspiring a paper section on data processing and statistical analysis. The multi-file approach could be adapted to real data, and the fitting routine could model detector effects or particle signals. The overlay plot is perfect for illustrating variations, such as momentum resolutions, and the fit table could summarize results.

D. Analyzing the ρ meson resonance

This project simulates an invariant mass distribution for 100,000 events, with 10 percent signal (Breit–Wigner fitted, mimicking ρ mesons) and 90 percent background (Gaussian fitted). This reflects HEP data storage from experiments related to particle decays. The concept of mixing signals received by the detector (identified by resonance peaks in the data) and background noise (identified by combinatorial or resolution effects) is central to resonance searches, such as identifying vector mesons in RHIC/STAR data. In other words, the choice of distributions (Gaussian + exponential vs. Breit-Wigner + Gaussian) and mass ranges reflect different physical scenarios and analysis goals.

```
abdallah_walid8@DESKTOP-JDADT5V:~$ root macro3.c
   ------------------------------------------------------------------
  | Welcome to ROOT 6.36.02                        https://root.cern |
  | (c) 1995-2025, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for linuxx8664gcc on Jul 16 2025, 09:29:01                 |
  | From tags/v6-36-02@v6-36-02                                      |
  | With c++ (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0                   |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0]
Processing macro3.c...
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
Gaussian Chi2/NDF (background): 0.389507
Breit-Wigner Chi2/NDF (signal): 1.14505
Info in <TCanvas::Print>: pdf file signal_background.pdf has been created
Info in <TCanvas::Print>: file signal_background.png has been created
Analysis complete. Separate signal and background stored.
root [1] .q
```



The data is read into a histogram and fitted separately with Gaussian (representing background) and Breit-Wigner (representing the signal) functions, demonstrating basic signal extraction. The fits, initialized near input parameters, yield high χ^2/ndf due to separate fitting of a mixed distribution, highlighting the need for combined models in real

analyses. The concept of fitting to extract resonance parameters such as mass, width, and yield is critical in HEP for quantifying particle production and testing physics models (such as QGP effects on ρ mesons). The plot overlays data points with error bars, the Gaussian fit as a blue line, and the Breit-Wigner fit as a red line, with a legend clarifying components. The histogram shows a broad background peak with a smaller signal peak. This visualization is key for HEP papers, illustrating signal significance and fit challenges.

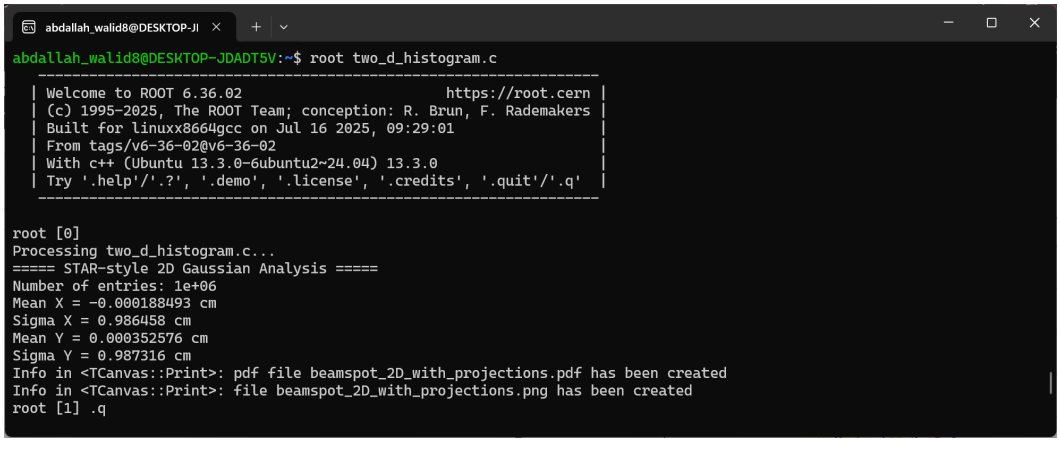
This code can inspire a paper section on signal extraction, with a figure showing the mass spectrum and a table of fit parameters. By implementing a combined signal+background fit (e.g., gaus+breitwigner), it is possible to extract yield and significance, or compare to theoretical predictions (e.g., ρ production in pp vs. Au-Au collisions). This workflow, summarized by the terms: generate, store, analyze, visualize—is a strong template for resonance studies.

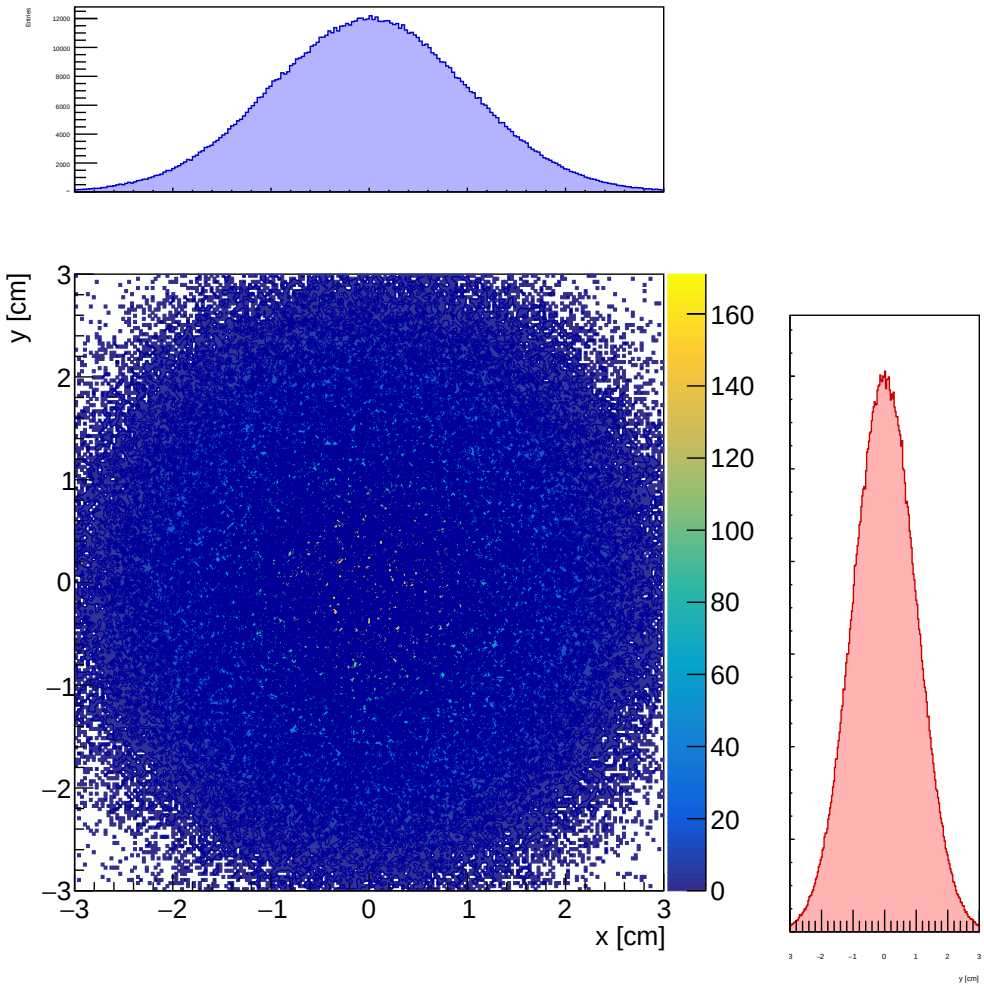
E. Creating a 2D histogram

This project simulates a two-dimensional Gaussian distribution in x and y with zero mean and unitary standard deviation. This can represent, for instance, the spatial distribution of a particle beam spot or detector response. A million events are generated and filled into a 2D histogram, with projections shown along each axis. This is a common method in collider physics to study beam alignment and detector resolution.

The 2D histogram reveals the overall shape of the distribution: a circular Gaussian cloud when x and y are independent. If the distribution were elliptical, it would indicate different resolutions along the axes or correlations between x and y. By computing covariance and correlation coefficients, one quantitatively checks for coupling or misalignment between the axes. In this simulation, the correlation is close to zero, confirming independence.

Projections onto x and y are fitted with Gaussian functions to extract precise estimates of mean and resolution for each coordinate. These fits provide uncertainties smaller than the raw RMS because they model the statistical distribution explicitly. This approach mirrors experimental procedures, where marginal distributions are fitted to quantify beam spot or vertex resolutions. The project also introduces advanced statistical ideas. The covariance matrix encodes both variances and correlations, and its diagonalization reveals principal axes (major/minor widths and orientation). This is the foundation of describing elliptical Gaussian distributions, widely used in accelerator physics and detector alignment. Thus, the code exemplifies both practical visualization (heatmaps, projections) and the theoretical framework for understanding multidimensional Gaussian statistics.





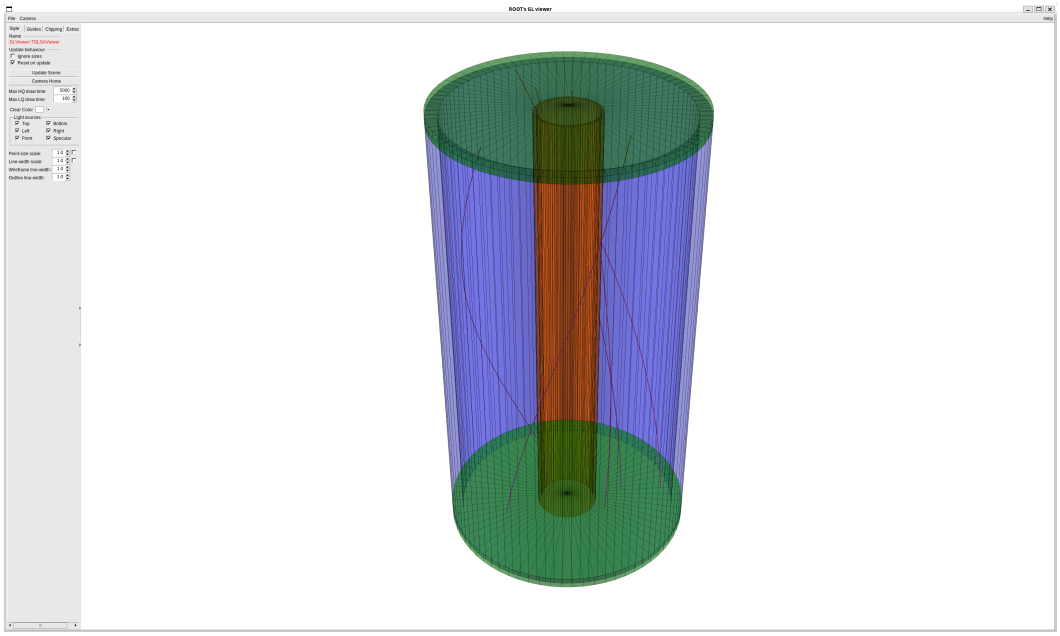
The two-dimensional spatial distribution of reconstructed vertices were simulated and analyzed to characterize the beam spot. A dataset of 1,000,000 events was generated with independent Gaussian distributions in x and y (μ=0, σ=1 cm). The event density was binned into a 200×200 two-dimensional histogram over −3 to 3 cm and plotted as a heatmap with contour overlays. One-dimensional projections onto x and y were fitted with Gaussian functions to extract mean positions and resolutions. The covariance matrix was computed from the sample to quantify correlations between axes, and principal axes were obtained by diagonalizing the covariance matrix. Statistical uncertainties on fitted parameters were obtained from the fit covariance matrices.

F. Creating a 3D visual

This project uses ROOT's TGeoManager to create a simplified 3D model of the STAR detector, including a TPC barrel (silicon tube), endcaps (aluminum disks), and an inner tracker (silicon layer). Defining materials (vacuum, aluminum, silicon) with physical properties (A, Z, density) and constructing hierarchical volumes (TGeoVolume) is a key concept, mirroring GEANT-based simulations for particle tracking. This geometry setup is essential for understanding detector interactions such as energy loss and scattering, and the simplified model here illustrates core components for visualization, critical for papers or presentations.

```
abdallah_walid8@DESKTOP-JI
abdallah_walid8@DESKTOP-JDADT5V:~$ root view2_3d.c
   ------------------------------------------------------------------
  | Welcome to ROOT 6.36.02                        https://root.cern |
  | (c) 1995-2025, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for linuxx8664gcc on Jul 16 2025, 09:29:01                 |
  | From tags/v6-36-02@v6-36-02                                      |
  | With c++ (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0                   |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0]
Processing view2_3d.c...
Info in <TGeoManager::TGeoManager>: Geometry STARgeom, STAR Detector Geometry created
Info in <TGeoManager::SetTopVolume>: Top volume is TOP. Master volume is TOP
Info in <TGeoNavigator::BuildCache>: --- Maximum geometry depth set to 100
Info in <TGeoManager::CheckGeometry>: Fixing runtime shapes...
Info in <TGeoManager::CheckGeometry>: ...Nothing to fix
Info in <TGeoManager::CloseGeometry>: Counting nodes...
Info in <TGeoManager::Voxelize>: Voxelizing...
Info in <TGeoManager::CloseGeometry>: Building cache...
Info in <TGeoManager::CountLevels>: max level = 1, max placements = 4
Info in <TGeoManager::CloseGeometry>: 5 nodes/ 4 volume UID's in STAR Detector Geometry
Info in <TGeoManager::CloseGeometry>: ----------------modeler ready----------------
Info in <TCanvas::Print>: file star_geometry_full.png has been created
Info in <TCanvas::Print>: pdf file star_geometry_full.pdf has been created
3D STAR-style geometry saved as PNG and PDF.
root [1] .q
```

The code generates five helical tracks to simulate charged particle trajectories in STAR's solenoidal magnetic field along the z-axis. Tracks spiral in the x-y plane (radius $\alpha p_T/B$) and advance linearly in z, with random starting angles and radii to mimic varying momenta. This concept is fundamental in HEP for tracking studies, where curvature yields momentum and vertex positions aid reconstruction, such as in heavy-flavor decays. Visualizing tracks within the geometry tests detector coverage, inspiring event-display figures in research papers. Using OpenGL (Draw("ogl")) for interactive 3D rendering, with outline style and camera rotation, provides a clear view of detector components and tracks. Transparency and color settings enhance visibility. This visualization technique is vital for illustrating detector design or simulation validation, often used in technical notes or conference talks to convey experimental setup.



This project can inspire a "Detector Simulation" section with a 3D figure showing STAR's layout or track reconstruction. It could be extended with real GEANT geometry or integrated with event generators to show realistic events. The figure is ideal for discussing acceptance or tracking efficiency, and the code's modularity allows adding more detector layers (such as calorimeters) for a more complete model, enhancing its academic value.

II. INTRODUCTION TO PYTHIA8

Pythia8 is mainly associated with event generation from first principles. Basically, it is a Monte Carlo event generator that simulates high-energy collisions between fundamental particles, such as those occurring at the Large Hadron Collider or the Relativistic Heavy Ion Collider. Pythia8 enables theorists and experimentalists to generate "truth-level" events, which serve as reference for detector simulations and the main validation tool used to validate or refute physics models. Its modular C++ structure, customizability, and compatibility with detector-level frameworks make it a foundational tool in both STAR and CERN research environments.

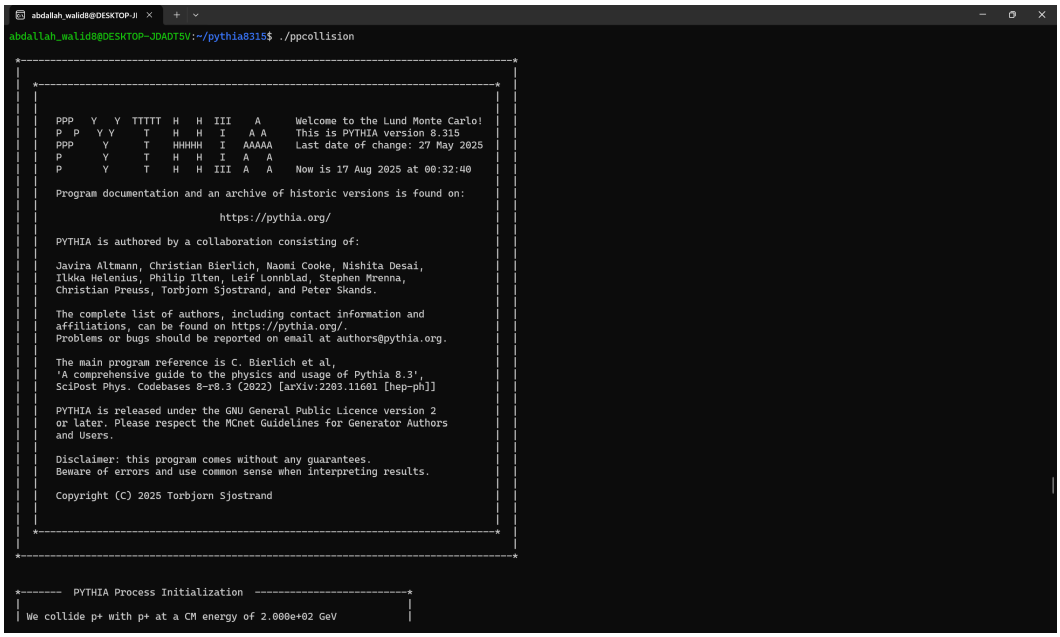
Proton-proton (pp) collisions occur when two protons are accelerated to high energies and smashed together in particle accelerators like the Large Hadron Collider at CERN or the Relativistic Heavy Ion Collider at Brookhaven. At everyday energies, protons behave as stable composite particles made of three valence quarks (two up and one down) bound by gluons, the carriers of the strong nuclear force described by Quantum Chromodynamics. However, in high-energy collisions (such as $\sqrt{s} = 13$ TeV at LHC or 200 GeV at RHIC), the protons' internal structure becomes relevant: they are clouds of quarks, antiquarks, and gluons (collectively called partons) whose distributions are encoded in Parton Distribution Functions. Collisions can be classified as elastic, where protons scatter intact, or inelastic, where protons break apart (producing new particles), or diffractive, where one or both protons remain intact but exchange momentum via Pomeron exchange, a QCD phenomenon. Inelastic collisions dominate at high energies, leading to a cascade of processes from initial parton interactions to final hadron production.

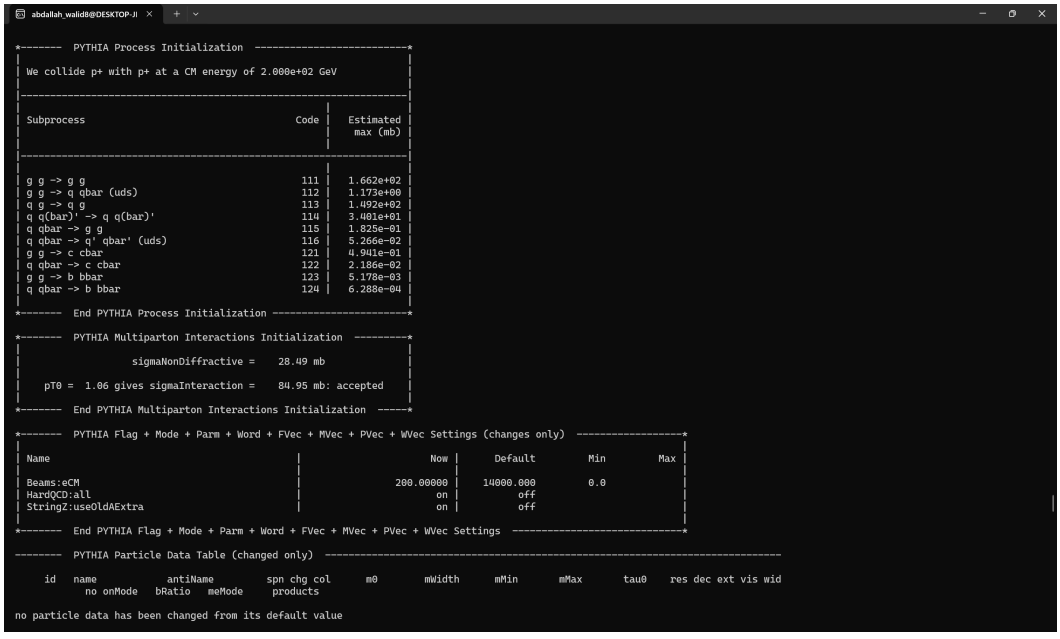
The total cross-section for pp collisions at LHC energies is around 100 millibarns, with QCD processes contributing the majority due to the strong force's dominance over electroweak interactions. Key observables include particle multiplicity (number of produced particles), transverse momentum (p_T) spectra, and correlations, which probe the proton's structure and QCD dynamics. These collisions serve as a baseline for studying heavy-ion physics, such as the quark-gluon plasma, and searching for new physics, but QCD effects like multiple parton interactions can create backgrounds that mimic rare signals. In contrast to soft QCD, hard processes involve high p_T ($>$ few GeV/c) where α_s is small (0.1−0.3), allowing pQCD calculations. These include jet production from 2→2 parton scatters (e.g., gg→gg, qg→qg). Parton showers add perturbative corrections, modeled as branching processes. Hard QCD probes α_s evolution and gluon PDFs, dominant at low x, where x is parton momentum fraction. At LHC, hard processes like inclusive jets or multijets are measured differentially, constraining PDFs and extracting α_s at Z-boson mass scale with NNLO accuracy. Factorization separates hard matrix elements from PDFs and showers, enabling predictions. Hard QCD constitutes backgrounds for new physics but also tests QCD at high scales, with some correlations being sensitive to new colored particles. Beyond QCD, electroweak processes occur, though with smaller cross-sections. These include Drell-Yan processes, W/Z production, and Higgs boson decays. QCD contributes to these via higher-order corrections (such as gluon radiation) and backgrounds (such as multijet fakes).

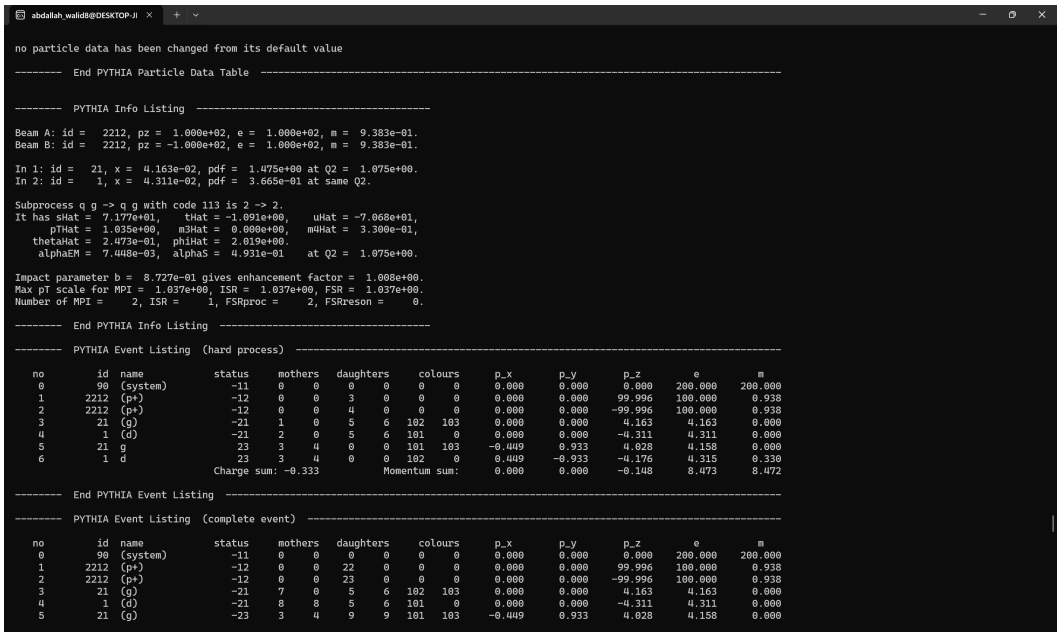
In pp collisions, new physics searches (e.g., supersymmetry, extra dimensions) rely on QCD modeling for backgrounds. Collective phenomena in high-multiplicity pp events suggest QGP-like effects, blurring lines with heavy-ion collisions, possibly from hydrodynamic flow or string shoving. Recent advances in QCD theory, such as reconciling quantum and relativistic aspects, improve proton structure understanding, linking collisions to internal dynamics via better data interpretation. Overall, pp collisions are a fundamental in QCD, with soft processes dominating bulk production and hard ones probing high scales, essential for precision physics at future colliders.

A. Proton-Proton collisions

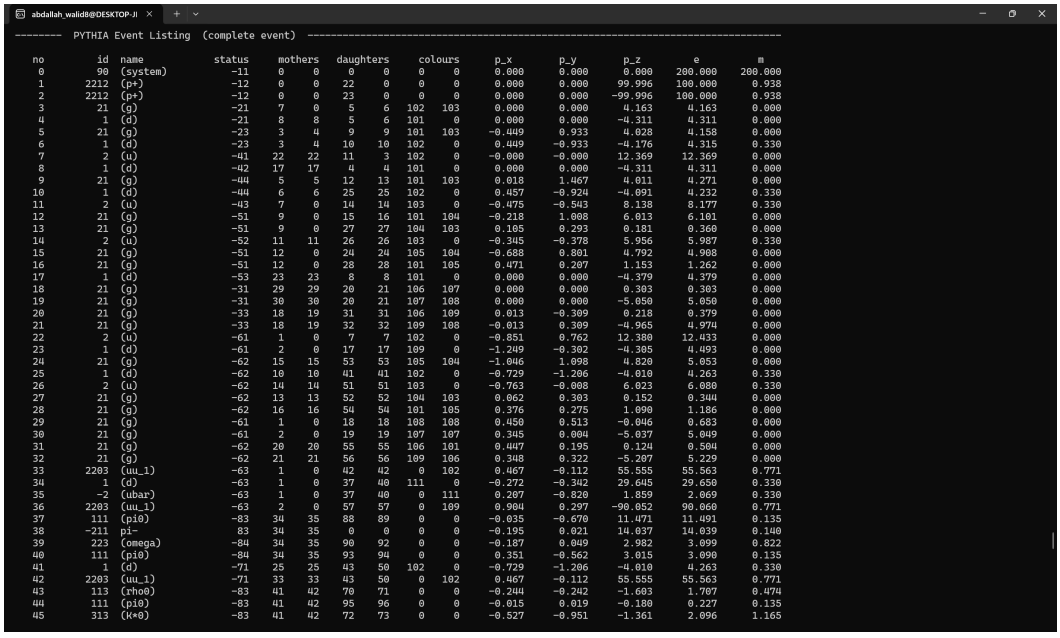
This project on proton-proton collisions code uses Pythia8, a Monte Carlo event generator, to simulate proton-proton (pp) collisions at 200 GeV, mimicking conditions at the Relativistic Heavy Ion Collider (RHIC) for the STAR experiment. A critical concept is the simulation of hard QCD processes (enabled via HardQCD:all = on), which involve high transverse momentum (p_T) parton scatterings (e.g., quark-gluon interactions) described by perturbative QCD (pQCD). Pythia8 models the full event evolution, including initial-state radiation, parton showers, multiple parton interactions, and hadronization via the Lund string model, transforming partons into observable hadrons. This allows researchers to predict particle spectra and test theoretical models against experimental data, a cornerstone of HEP simulation workflows.



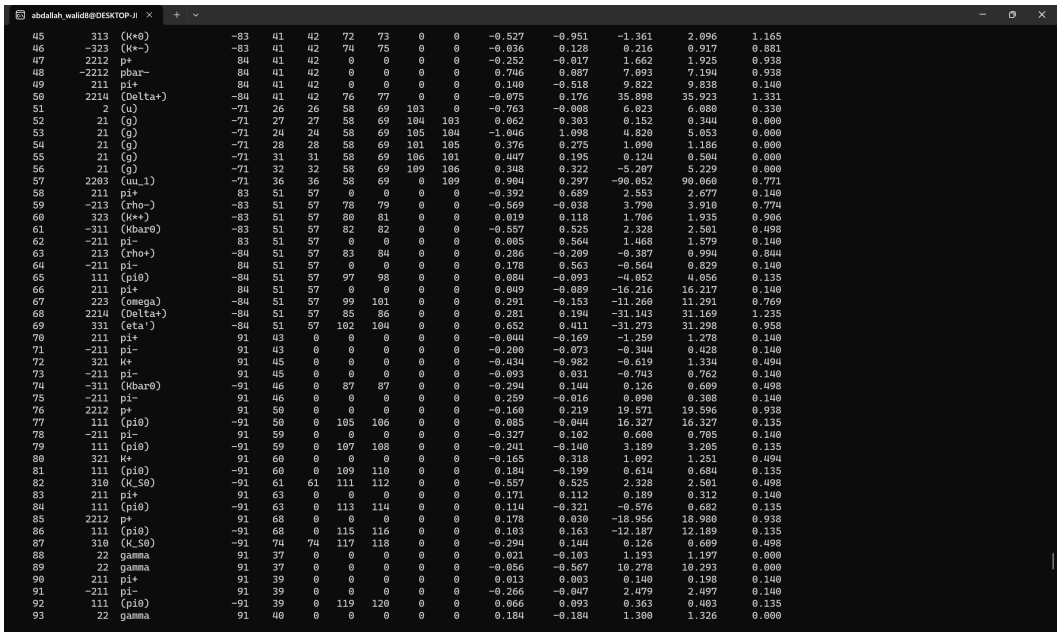




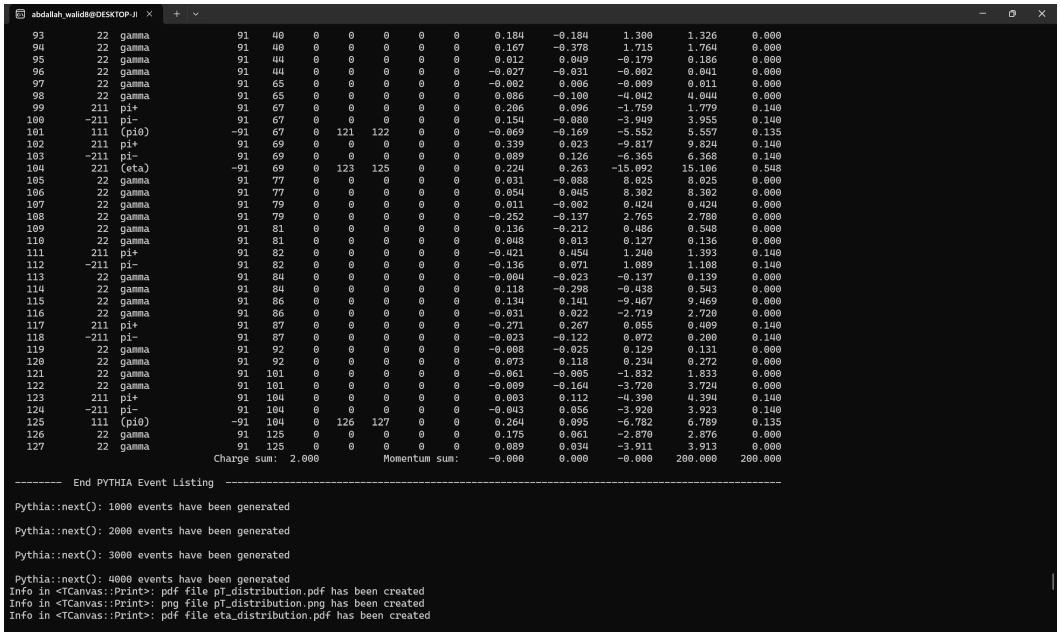
```
abdallah_walid8@DESKTOP-JI

no particle data has been changed from its default value

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

--------  PYTHIA Info Listing  ------------------------------------

Beam A: id =   2212, pz =  1.000e+02, e =  1.000e+02, m =  9.383e-01.
Beam B: id =   2212, pz = -1.000e+02, e =  1.000e+02, m =  9.383e-01.

In 1: id =   21, x =  4.163e-02, pdf =  1.475e+00 at Q2 =  1.075e+00.
In 2: id =    1, x =  4.311e-02, pdf =  3.665e-01 at same Q2.

Subprocess q g -> q g with code 113 is 2 -> 2.
It has sHat =  7.177e+01,    tHat = -1.091e+00,    uHat = -7.068e+01,
      pTHat =  1.035e+00,   m3Hat =  0.000e+00,   m4Hat =  3.300e-01,
   thetaHat =  2.473e-01,  phiHat =  2.019e+00.
    alphaEM =  7.448e-03,  alphaS =  4.931e-01    at Q2 =  1.075e+00.

Impact parameter b =  8.727e-01 gives enhancement factor =  1.008e+00.
Max pT scale for MPI =  1.037e+00, ISR =  1.037e+00, FSR =  1.037e+00.
Number of MPI =     2, ISR =     1, FSRproc =     2, FSRreson =     0.

--------  End PYTHIA Info Listing  --------------------------------

--------  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     0      0.000      0.000      0.000    200.000    200.000
    1       2212  (p+)               -12     0     0     3     0     0     0      0.000      0.000     99.996    100.000      0.938
    2       2212  (p+)               -12     0     0     4     0     0     0      0.000      0.000    -99.996    100.000      0.938
    3         21  (g)                -21     1     0     5     6   102   103      0.000      0.000      4.163      4.163      0.000
    4          1  (d)                -21     2     0     5     6   101     0      0.000      0.000     -4.311      4.311      0.000
    5         21  g                   23     3     4     0     0   101   103     -0.449      0.933      4.028      4.158      0.000
    6          1  d                   23     3     4     0     0   102     0      0.449     -0.933     -4.176      4.315      0.330
                                 Charge sum: -0.333          Momentum sum:      0.000      0.000     -0.148      8.473      8.472

--------  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     0      0.000      0.000      0.000    200.000    200.000
    1       2212  (p+)               -12     0     0    22     0     0     0      0.000      0.000     99.996    100.000      0.938
    2       2212  (p+)               -12     0     0    23     0     0     0      0.000      0.000    -99.996    100.000      0.938
    3         21  (g)                -21     7     0     5     6   102   103      0.000      0.000      4.163      4.163      0.000
    4          1  (d)                -21     8     8     5     6   101     0      0.000      0.000     -4.311      4.311      0.000
    5         21  (g)                -23     3     4     9     9   101   103     -0.449      0.933      4.028      4.158      0.000
```



```
abdallah_walid8@DESKTOP-JI

--------  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     0      0.000      0.000      0.000    200.000    200.000
    1       2212  (p+)               -12     0     0    22     0     0     0      0.000      0.000     99.996    100.000      0.938
    2       2212  (p+)               -12     0     0    23     0     0     0      0.000      0.000    -99.996    100.000      0.938
    3         21  (g)                -21     7     0     5     6   102   103      0.000      0.000      4.163      4.163      0.000
    4          1  (d)                -21     8     8     5     6   101     0      0.000      0.000     -4.311      4.311      0.000
    5         21  (g)                -23     3     4     9     9   101   103     -0.449      0.933      4.028      4.158      0.000
    6          1  (d)                -23     3     4    10    10   102     0      0.449     -0.933     -4.176      4.315      0.330
    7          2  (u)                -41    22    22    11     3   102     0     -0.000     -0.000     12.369     12.369      0.000
    8          1  (d)                -42    17    17     4     4   101     0      0.000      0.000     -4.311      4.311      0.000
    9         21  (g)                -44     5     5    12    13   101   103      0.018      1.467      4.011      4.271      0.000
   10          1  (d)                -44     6     6    25    25   102     0      0.457     -0.924     -4.091      4.232      0.330
   11          2  (u)                -43     7     0    14    14   103     0     -0.475     -0.543      8.138      8.177      0.330
   12         21  (g)                -51     9     0    15    16   101   104     -0.218      1.008      6.013      6.101      0.000
   13         21  (g)                -51     9     0    27    27   104   103      0.105      0.293      0.181      0.360      0.000
   14          2  (u)                -52    11    11    26    26   103     0     -0.345     -0.378      5.956      5.987      0.330
   15         21  (g)                -51    12     0    24    24   105   104     -0.688      0.801      4.792      4.908      0.000
   16         21  (g)                -51    12     0    28    28   101   105      0.471      0.207      1.153      1.262      0.000
   17          1  (d)                -53    23    23     8     8   101     0      0.000      0.000     -4.379      4.379      0.000
   18         21  (g)                -31    29    29    20    21   106   107      0.000      0.000      0.303      0.303      0.000
   19         21  (g)                -31    30    30    20    21   107   108      0.000      0.000     -5.050      5.050      0.000
   20         21  (g)                -33    18    19    31    31   106   109      0.013     -0.309      0.218      0.379      0.000
   21         21  (g)                -33    18    19    32    32   109   108     -0.013      0.309     -4.965      4.974      0.000
   22          2  (u)                -61     1     0     7     7   102     0     -0.851      0.762     12.380     12.433      0.000
   23          1  (d)                -61     2     0    17    17   109     0     -1.249     -0.302     -4.305      4.493      0.000
   24         21  (g)                -62    15    15    53    53   105   104     -1.046      1.098      4.820      5.053      0.000
   25          1  (d)                -62    10    10    41    41   102     0     -0.729     -1.206     -4.010      4.263      0.330
   26          2  (u)                -62    14    14    51    51   103     0     -0.763     -0.008      6.023      6.080      0.330
   27         21  (g)                -62    13    13    52    52   104   103      0.062      0.303      0.152      0.344      0.000
   28         21  (g)                -62    16    16    54    54   101   105      0.376      0.275      1.090      1.186      0.000
   29         21  (g)                -61     1     0    18    18   108   108      0.450      0.513     -0.046      0.683      0.000
   30         21  (g)                -61     2     0    19    19   107   107      0.345      0.004     -5.037      5.049      0.000
   31         21  (g)                -62    20    20    55    55   106   101      0.447      0.195      0.134      0.504      0.000
   32         21  (g)                -62    21    21    56    56   109   106      0.348      0.322     -5.207      5.229      0.000
   33       2203  (uu_1)             -63     1     0    42    42     0   102      0.467     -0.112     55.555     55.563      0.771
   34          1  (d)                -63     1     0    37    40   111     0     -0.272     -0.342     29.645     29.650      0.330
   35         -2  (ubar)             -63     1     0    37    40     0   111      0.207     -0.820      1.859      2.069      0.330
   36       2203  (uu_1)             -63     2     0    57    57     0   109      0.904      0.297    -90.052     90.060      0.771
   37        111  (pi0)              -83    34    35    88    89     0     0     -0.035     -0.670     11.471     11.491      0.135
   38       -211  pi-                83    34    35     0     0     0     0     -0.195      0.021     14.037     14.039      0.140
   39        223  (omega)            -84    34    35    90    92     0     0     -0.187      0.049      2.982      3.099      0.822
   40        111  (pi0)              -84    34    35    93    94     0     0      0.351     -0.562      3.015      3.090      0.135
   41          1  (d)                -71    25    25    43    50   102     0     -0.729     -1.206     -4.010      4.263      0.330
   42       2203  (uu_1)             -71    33    33    43    50     0   102      0.467     -0.112     55.555     55.563      0.771
   43        113  (rho0)             -83    41    42    70    71     0     0     -0.244     -0.242     -1.603      1.707      0.474
   44        111  (pi0)              -83    41    42    95    96     0     0     -0.015      0.019     -0.180      0.227      0.135
   45        313  (K*0)              -83    41    42    72    73     0     0     -0.527     -0.951     -1.361      2.096      1.165
```

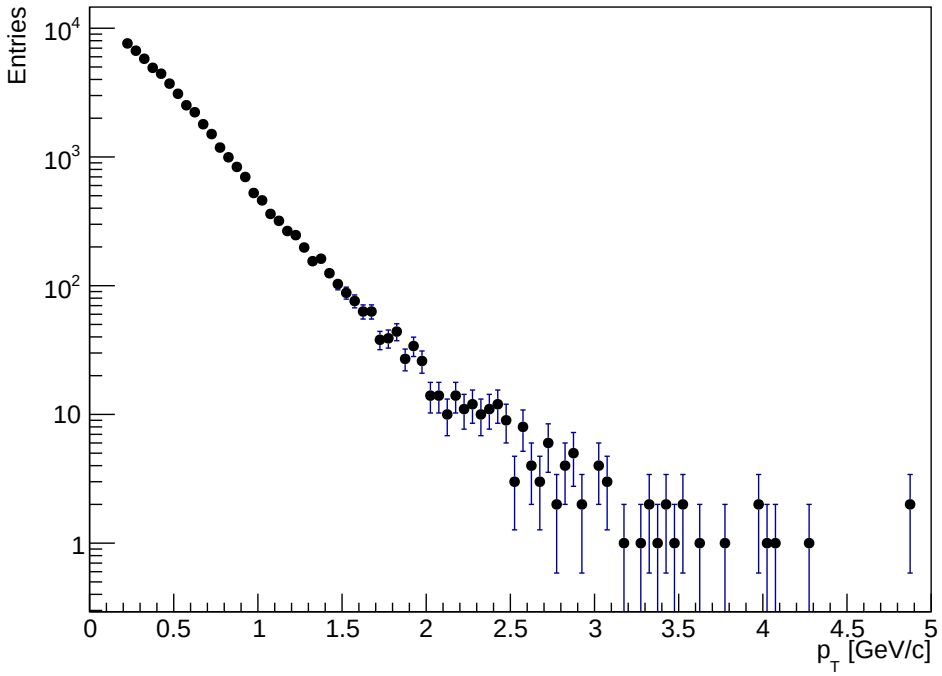


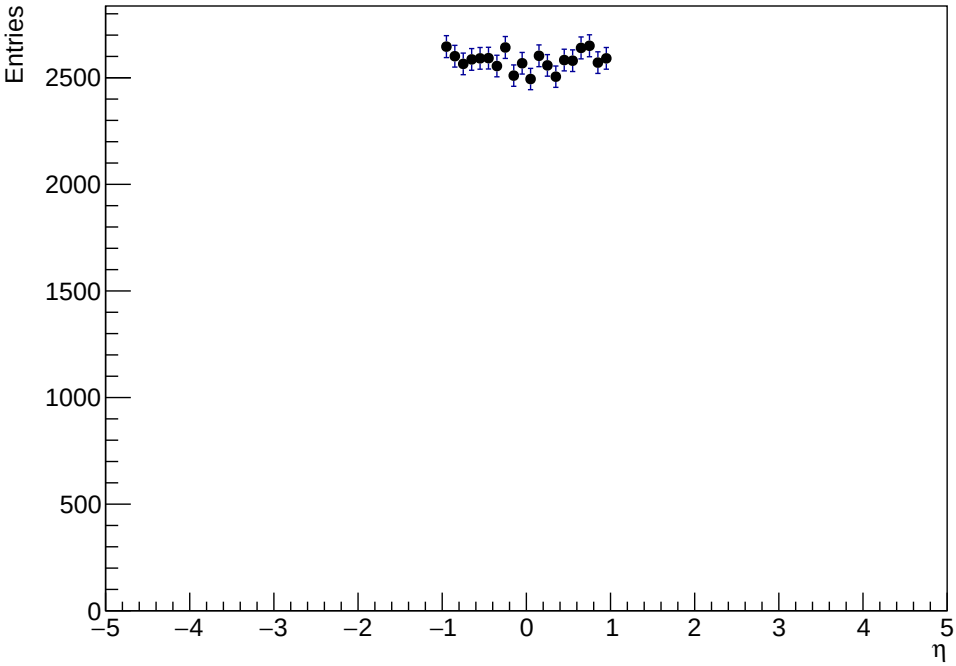
```
abdallah_walid8@DESKTOP-JI

   45        313  (K*0)            -83    41    42    72    73     0     0    -0.527    -0.951    -1.361     2.096     1.165
   46       -323  (K*-)            -83    41    42    74    75     0     0    -0.036     0.128     0.216     0.917     0.881
   47       2212  p+                84    41    42     0     0     0     0    -0.252    -0.017     1.662     1.925     0.938
   48      -2212  pbar-             84    41    42     0     0     0     0     0.746     0.087     7.093     7.194     0.938
   49        211  pi+               84    41    42     0     0     0     0     0.140    -0.518     9.822     9.838     0.140
   50       2214  (Delta+)          84    41    42    76    77     0     0    -0.075     0.176    35.898    35.923     1.331
   51          2  (u)              -71    26    26    58    69   103     0    -0.763    -0.008     6.023     6.080     0.330
   52         21  (g)              -71    27    27    58    69   104   103     0.062     0.303     0.152     0.344     0.000
   53         21  (g)              -71    24    24    58    69   105   104    -1.046     1.098     4.820     5.053     0.000
   54         21  (g)              -71    28    28    58    69   101   105     0.376     0.275     1.090     1.186     0.000
   55         21  (g)              -71    31    31    58    69   106   101     0.447     0.195     0.124     0.504     0.000
   56         21  (g)              -71    32    32    58    69   109   106     0.348     0.322    -5.207     5.229     0.000
   57       2203  (uu_1)           -71    36    36    58    69     0   109     0.904     0.297   -90.052    90.060     0.771
   58        211  pi+               83    51    57     0     0     0     0    -0.392     0.689     2.553     2.677     0.140
   59       -213  (rho-)           -83    51    57    78    79     0     0    -0.569    -0.038     3.790     3.910     0.774
   60        323  (K*+)            -83    51    57    80    81     0     0     0.019     0.118     1.706     1.935     0.906
   61       -311  (Kbar0)          -83    51    57    82    82     0     0    -0.557     0.525     2.328     2.501     0.498
   62       -211  pi-               83    51    57     0     0     0     0     0.005     0.564     1.468     1.579     0.140
   63        213  (rho+)           -84    51    57    83    84     0     0     0.286    -0.209    -0.387     0.994     0.844
   64       -211  pi-               84    51    57     0     0     0     0     0.178     0.563    -0.564     0.829     0.140
   65        111  (pi0)            -84    51    57    97    98     0     0     0.084    -0.093    -4.052     4.056     0.135
   66        211  pi+               84    51    57     0     0     0     0     0.049    -0.089   -16.216    16.217     0.140
   67        223  (omega)          -84    51    57    99   101     0     0     0.291    -0.153   -11.260    11.291     0.769
   68       2214  (Delta+)         -84    51    57    85    86     0     0     0.281     0.194   -31.143    31.169     1.235
   69        331  (eta')           -84    51    57   102   104     0     0     0.652     0.411   -31.273    31.298     0.958
   70        211  pi+               91    43     0     0     0     0     0    -0.044    -0.169    -1.259     1.278     0.140
   71       -211  pi-               91    43     0     0     0     0     0    -0.200    -0.073    -0.344     0.428     0.140
   72        321  K+                91    45     0     0     0     0     0    -0.434    -0.982    -0.619     1.334     0.494
   73       -211  pi-               91    45     0     0     0     0     0    -0.093     0.031    -0.743     0.762     0.140
   74       -311  (Kbar0)          -91    46     0    87    87     0     0    -0.294     0.144     0.126     0.609     0.498
   75       -211  pi-               91    46     0     0     0     0     0     0.259    -0.016     0.090     0.308     0.140
   76       2212  p+                91    50     0     0     0     0     0    -0.160     0.219    19.571    19.596     0.938
   77        111  (pi0)            -91    50     0   105   106     0     0     0.085    -0.044    16.327    16.327     0.135
   78       -211  pi-               91    59     0     0     0     0     0    -0.327     0.102     0.600     0.705     0.140
   79        111  (pi0)            -91    59     0   107   108     0     0    -0.241    -0.140     3.189     3.205     0.135
   80        321  K+                91    60     0     0     0     0     0    -0.165     0.318     1.092     1.251     0.494
   81        111  (pi0)            -91    60     0   109   110     0     0     0.184    -0.199     0.614     0.684     0.135
   82        310  (K_S0)           -91    61    61   111   112     0     0    -0.557     0.525     2.328     2.501     0.498
   83        211  pi+               91    63     0     0     0     0     0     0.171     0.112     0.189     0.312     0.140
   84        111  (pi0)            -91    63     0   113   114     0     0     0.114    -0.321    -0.576     0.682     0.135
   85       2212  p+                91    68     0     0     0     0     0     0.178     0.030   -18.956    18.980     0.938
   86        111  (pi0)            -91    68     0   115   116     0     0     0.103     0.163   -12.187    12.189     0.135
   87        310  (K_S0)           -91    74    74   117   118     0     0    -0.294     0.144     0.126     0.609     0.498
   88         22  gamma             91    37     0     0     0     0     0     0.021    -0.103     1.193     1.197     0.000
   89         22  gamma             91    37     0     0     0     0     0    -0.056    -0.567    10.278    10.293     0.000
   90        211  pi+               91    39     0     0     0     0     0     0.013     0.003     0.140     0.198     0.140
   91       -211  pi-               91    39     0     0     0     0     0    -0.266    -0.047     2.479     2.497     0.140
   92        111  (pi0)            -91    39     0   119   120     0     0     0.066     0.093     0.363     0.403     0.135
   93         22  gamma             91    40     0     0     0     0     0     0.184    -0.184     1.300     1.326     0.000
```

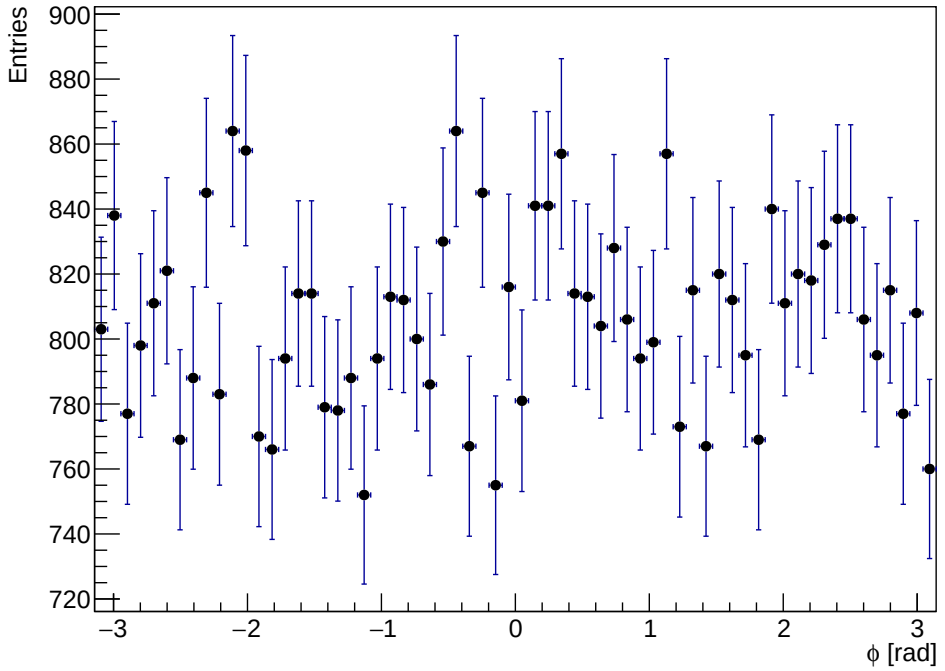


```
abdallah_walid8@DESKTOP-JI

   93         22  gamma             91    40     0     0     0     0     0     0.184    -0.184     1.300     1.326     0.000
   94         22  gamma             91    40     0     0     0     0     0     0.167    -0.378     1.715     1.764     0.000
   95         22  gamma             91    44     0     0     0     0     0     0.012     0.049    -0.179     0.186     0.000
   96         22  gamma             91    44     0     0     0     0     0    -0.027    -0.031    -0.002     0.041     0.000
   97         22  gamma             91    65     0     0     0     0     0    -0.002     0.006    -0.009     0.011     0.000
   98         22  gamma             91    65     0     0     0     0     0     0.086    -0.100    -4.042     4.044     0.000
   99        211  pi+               91    67     0     0     0     0     0     0.206     0.096    -1.759     1.779     0.140
  100       -211  pi-               91    67     0     0     0     0     0     0.154    -0.080    -3.949     3.955     0.140
  101        111  (pi0)            -91    67     0   121   122     0     0    -0.069    -0.169    -5.552     5.557     0.135
  102        211  pi+               91    69     0     0     0     0     0     0.339     0.023    -9.817     9.824     0.140
  103       -211  pi-               91    69     0     0     0     0     0     0.089     0.126    -6.365     6.368     0.140
  104        221  (eta)            -91    69     0   123   125     0     0     0.224     0.263   -15.092    15.106     0.548
  105         22  gamma             91    77     0     0     0     0     0     0.031    -0.088     8.026     8.025     0.000
  106         22  gamma             91    77     0     0     0     0     0     0.054     0.045     8.302     8.302     0.000
  107         22  gamma             91    79     0     0     0     0     0     0.011    -0.002     0.424     0.424     0.000
  108         22  gamma             91    79     0     0     0     0     0    -0.252    -0.137     2.765     2.780     0.000
  109         22  gamma             91    81     0     0     0     0     0     0.136    -0.212     0.486     0.548     0.000
  110         22  gamma             91    81     0     0     0     0     0     0.048     0.013     0.127     0.136     0.000
  111        211  pi+               91    82     0     0     0     0     0    -0.421     0.454     1.240     1.393     0.140
  112       -211  pi-               91    82     0     0     0     0     0    -0.136     0.071     1.089     1.108     0.140
  113         22  gamma             91    84     0     0     0     0     0    -0.004    -0.023    -0.137     0.139     0.000
  114         22  gamma             91    84     0     0     0     0     0     0.118    -0.298    -0.438     0.543     0.000
  115         22  gamma             91    86     0     0     0     0     0     0.134     0.141    -9.467     9.469     0.000
  116         22  gamma             91    86     0     0     0     0     0    -0.031     0.022    -2.719     2.720     0.000
  117        211  pi+               91    87     0     0     0     0     0    -0.271     0.267     0.055     0.409     0.140
  118       -211  pi-               91    87     0     0     0     0     0    -0.023    -0.122     0.072     0.200     0.140
  119         22  gamma             91    92     0     0     0     0     0    -0.008    -0.025     0.129     0.131     0.000
  120         22  gamma             91    92     0     0     0     0     0     0.073     0.118     0.234     0.272     0.000
  121         22  gamma             91   101     0     0     0     0     0    -0.061    -0.005    -1.832     1.833     0.000
  122         22  gamma             91   101     0     0     0     0     0    -0.009    -0.164    -3.720     3.724     0.000
  123        211  pi+               91   104     0     0     0     0     0     0.003     0.112    -4.390     4.394     0.140
  124       -211  pi-               91   104     0     0     0     0     0    -0.043     0.056    -3.920     3.923     0.140
  125        111  (pi0)            -91   104     0   126   127     0     0     0.264     0.095    -6.782     6.789     0.135
  126         22  gamma             91   125     0     0     0     0     0     0.175     0.061    -2.870     2.876     0.000
  127         22  gamma             91   125     0     0     0     0     0     0.089     0.034    -3.911     3.913     0.000
                                Charge sum:  2.000          Momentum sum:    -0.000     0.000    -0.000   200.000   200.000

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

 Pythia::next(): 1000 events have been generated

 Pythia::next(): 2000 events have been generated

 Pythia::next(): 3000 events have been generated

 Pythia::next(): 4000 events have been generated
Info in <TCanvas::Print>: pdf file pT_distribution.pdf has been created
Info in <TCanvas::Print>: png file pT_distribution.png has been created
Info in <TCanvas::Print>: pdf file eta_distribution.pdf has been created
```

```
abdallah_walid8@DESKTOP-JI

 Pythia::next(): 1000 events have been generated

 Pythia::next(): 2000 events have been generated

 Pythia::next(): 3000 events have been generated

 Pythia::next(): 4000 events have been generated
Info in <TCanvas::Print>: pdf file pT_distribution.pdf has been created
Info in <TCanvas::Print>: png file pT_distribution.png has been created
Info in <TCanvas::Print>: pdf file eta_distribution.pdf has been created
Info in <TCanvas::Print>: png file eta_distribution.png has been created
Info in <TCanvas::Print>: pdf file phi_distribution.pdf has been created
Info in <TCanvas::Print>: png file phi_distribution.png has been created
Info in <TCanvas::Print>: pdf file pT_vs_eta.pdf has been created
Info in <TCanvas::Print>: png file pT_vs_eta.png has been created

 *-------  PYTHIA Event and Cross Section Statistics  -------------------------------------------------*
 |                                                                                                     |
 | Subprocess                                    Code |            Number of events        |      sigma +- delta    |
 |                                                    |       Tried   Selected   Accepted  |     (estimated) (mb)   |
 |                                                    |                                    |                        |
 |-----------------------------------------------------------------------------------------------------|
 |                                                    |                                    |                        |
 | g g -> g g                                      111 |       12335       2412       2412 |   3.207e+01  3.714e-01 |
 | g g -> q qbar (uds)                             112 |         100         25         25 |   3.179e-01  3.266e-02 |
 | q g -> q g                                      113 |       11113       2143       2143 |   2.887e+01  3.704e-01 |
 | q q(bar)' -> q q(bar)'                          114 |        2514        413        413 |   5.509e+00  1.424e-01 |
 | q qbar -> g g                                   115 |           9          1          1 |   3.817e-02  3.817e-02 |
 | q qbar -> q' qbar' (uds)                        116 |           3          1          1 |   1.409e-02  1.409e-02 |
 | g g -> c cbar                                   121 |          31          5          5 |   7.538e-02  2.247e-02 |
 | q qbar -> c cbar                                122 |           1          0          0 |   0.000e+00  0.000e+00 |
 | g g -> b bbar                                   123 |           2          0          0 |   0.000e+00  0.000e+00 |
 | q qbar -> b bbar                                124 |           0          0          0 |   0.000e+00  0.000e+00 |
 |                                                    |                                    |                        |
 | sum                                                |       26108       5000       5000 |   6.689e+01  5.465e-01 |
 |                                                    |                                    |                        |
 *-------  End PYTHIA Event and Cross Section Statistics -------------------------------------------------*

 *-------  PYTHIA Error and Warning Messages Statistics  ----------------------------------------------*
 |                                                                                                     |
 |  times   message                                                                                    |
 |                                                                                                     |
 |      0   no errors or warnings to report                                                            |
 |                                                                                                     |
 *-------  End PYTHIA Error and Warning Messages Statistics  -------------------------------------------*
abdallah_walid8@DESKTOP-JDADT5V:~/pythia8315$
```

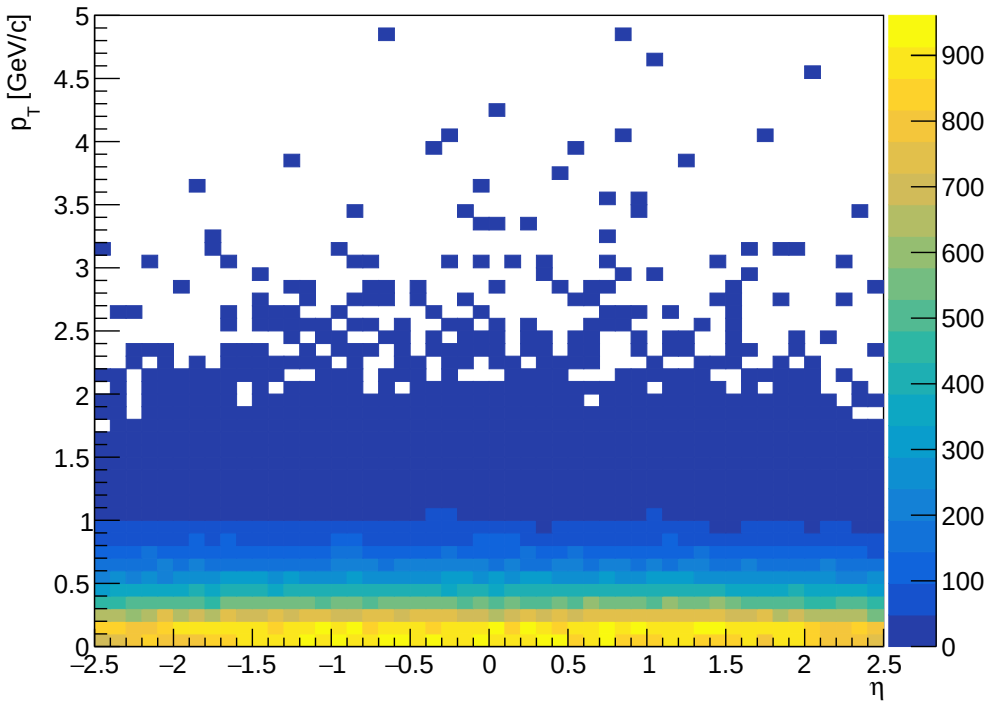






The code generates histograms for key kinematic variables—p_T (transverse momentum), η (pseudorapidity), and ϕ (azimuthal angle)—using ROOT's histogram classes (TH1F, TH2F). A crucial concept is the application of detector acceptance cuts ($|\eta| < 1$, $p_T > 0.2$ GeV/c), reflecting STAR's central tracking capabilities. These cuts filter particles to those detectable by the experiment, enabling realistic comparisons with data. The histograms

capture distributions: p_T follows a steep exponential (soft) to power-law (hard) shape, η is flat in the central rapidity plateau, and ϕ is uniform due to cylindrical symmetry. The 2D η vs p_T histogram below shows full kinematic coverage, highlighting detector limitations, which is vital for understanding experimental biases in papers.



ROOT's visualization tools can produce publication-quality plots with logarithmic axes for p_T (to emphasize tails) and a color map for 2D plots. The use of Sumw2() ensures proper statistical error propagation (Poisson uncertainties), critical for HEP analyses where precision matters. The code's output, including Pythia statistics, provides insights into process contributions (e.g., gg→gg dominance). This workflow—simulate, filter, histogram, visualize—is standard for validating Monte Carlo predictions against data, inspiring sections in papers on simulation methods or kinematic results.

This project exemplifies how to simulate collider events, apply experimental constraints, and analyze distributions. It could inspire a methodology section detailing Monte Carlo tools, a results section with kinematic plots, or comparisons with STAR data (e.g., p_T spectra fitted with Tsallis functions). Extending the code by adding particle identification or soft QCD processes could enhance its applicability to specific physics questions, such as jet production or bulk properties in pp collisions.

III. REFERENCES

[1]ROOT Team, *ROOT Data Analysis Framework: Users Guide* (2024), `https://root.cern/manual/`.
[2]ROOT Team, *ROOT Class Reference* (2024), `https://root.cern/doc/master/`.
[3]G. Cowan, *Statistical Data Analysis* (Oxford University Press, 1998).
[4]L. Lyons, *Statistics for Nuclear and Particle Physicists* (Cambridge University Press, 1986).
[5]Particle Data Group, "Review of particle physics," Prog. Theor. Exp. Phys. **2024**, 083C01 (2024), `https://pdg.lbl.gov/`.
[6]R. Ellis, W. Stirling, and B. Webber, "Qcd and collider physics," in *Cambridge Monographs on Particle Physics, Nuclear Physics and Cosmology* (Cambridge University Press, 1996).
[7]CMS Collaboration, "Top quark physics at the lhc," arXiv preprint (2014), arXiv:1401.6075.
[8]C. Heinz, *The Listings Package* (2023), `https://ctan.org/pkg/listings`.
[9]U. Kern, *The xcolor package* (2023), `https://ctan.org/pkg/xcolor`.
[10]J. Ellis, *tikz-feynman: Feynman diagrams with TikZ* (2023), `https://ctan.org/pkg/tikz-feynman`.
[11]Wikibooks contributors, "Latex wikibook," (2024), `https://en.wikibooks.org/wiki/LaTeX`.
[12]B. Frieden, *Probability, Statistical Optics, and Data Testing* (Springer, 2001).
[13]CMS Collaboration, "Measurement of the cms beam spot during lhc run 2," Tech. Rep. CMS-DP-2016-070 (CERN, 2016).
[14]C. Collaboration, "Measurement of the underlying event activity in proton-proton collisions at 13 tev with the cms detector," Journal of High Energy Physics **2015**, 008 (2015), available at `https://arxiv.org/abs/1512.00815`, arXiv:1512.00815.
[15]T. Sjöstrand, S. Mrenna, and P. Skands, "A brief introduction to pythia 8.1," Computer Physics Communications **178**, 852–867 (2008).
[16]A. Collaboration, "Enhanced production of multi-strange hadrons in high-multiplicity proton-proton collisions," Nature Physics **13**, 535–539 (2017).
[17]H1 and Z. Collaborations, "Combination of hera deep inelastic scattering data and impact on pdfs," European Physical Journal C **80**, 118 (2020).
[18]S. Collaboration, "Charged particle distributions in pp collisions at $\sqrt{s} = 200$ gev," Physical Review D **74**, 032006 (2005).
[19]R. Brun and F. Rademakers, "Root — an object-oriented data analysis framework," Nuclear Instruments and Methods in Physics Research Section A **389**, 81–86 (1997).
[20]P. Skands *et al.*, "Qcd for collider physics," Annual Review of Nuclear and Particle Science **64**, 159–186 (2014).
[21]R. Barlow, *Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences* (John Wiley & Sons, 2002).
[22]C. Collaboration, "Measurement of charged particle spectra in pp collisions at $\sqrt{s} = 13$ tev," Journal of High Energy Physics **2018**, 130 (2018).
[23]S. Collaboration, "The star time projection chamber: A unique tool for studying high multiplicity events at rhic," Nuclear Instruments and Methods in Physics Research Section A **499**, 659–678 (2003).
[24]S. Agostinelli *et al.*, "Geant4—a simulation toolkit," Nuclear Instruments and Methods in Physics Research Section A **506**, 250–303 (2003).
[25]R. Brun *et al.*, "Root geometry package," in *Proceedings of CHEP 2003* (2003) available via CERN Document Server or ROOT documentation: `https://root.cern/doc/master/`.
[26]S. Collaboration, "ρ^0 production and possible modification in au+au and pp collisions at $\sqrt{s_{NN}} = 200$ gev," Physical Review Letters **99**, 112301 (2007).
[27]M. E. Peskin and D. V. Schroeder, *An Introduction to Quantum Field Theory* (Addison-Wesley, 1995).

Appendix A: Codes

This appendix shows all the codes used for each project, with comments added to improve the reader's understanding of the logic and flow of the codes.

1. Importing randomly generated data into a simple histogram

Listing 1. extractandplot.c

```
void extractandplot()
{

    //Create histogram and random number generator
    TH1F *hist = new TH1F("hist", "", 100, 0, 15);   // Histogram
        with 100 bins, range 0  15
    TRandom2 *rand = new TRandom2(3);                // Random
        generator with seed=3
    fstream file;

    // Generate random Gaussian-distributed data and save to file
    file.open("data.txt", ios::out);
    for (int i = 0; i < 1000; i++)
    {
        double r = rand->Gaus(5, 1);   // Gaussian with mean=5, sigma
            =1
        file << r << endl;             // Write values to file
    }
    file.close();


    // Read data back from file and fill histogram
    file.open("data.txt", ios::in);
    double value;
    while (1)
    {
        file >> value;
        hist->Fill(value);             // Fill histogram with each
            value
        if (file.eof()) break;         // Stop at end of file
    }
    file.close();

    // Customize histogram appearance
    hist->SetFillColor(kGreen - 9);
    hist->GetXaxis()->SetTitle("Distribution");
    hist->GetYaxis()->SetTitle("Entries");
    hist->GetXaxis()->SetTitleSize(0.05);
    hist->GetYaxis()->SetTitleSize(0.05);
    hist->GetXaxis()->SetLabelSize(0.05);
    hist->GetYaxis()->SetLabelSize(0.05);


    // Define Gaussian fit function with initial parameters
    TF1 *fit = new TF1("fit", "gaus", 0, 15);
    fit->SetLineWidth(3);
    fit->SetLineColor(kRed);
    fit->SetLineStyle(2);
    fit->SetParameter(0, 40);    // Normalization (initial guess)
    fit->SetParameter(1, 5);     // Mean (initial guess)
```

```
    fit->SetParameter(2, 1);    // Sigma (initial guess)


    // Create canvas, enable grid, fit histogram with Gaussian, and
        draw
    TCanvas *c1 = new TCanvas();
    c1->SetTickx();
    c1->SetTicky();
    c1->SetGridx();
    c1->SetGridy();
    hist->Fit("fit");   // Perform Gaussian fit
    hist->Draw();       // Draw histogram

    // Add legend to distinguish data and fit
    TLegend *leg = new TLegend(0.5, 0.55, 0.8, 0.75);
    leg->SetBorderSize(0);
    leg->AddEntry(hist, "Measured Data", "p");
    leg->AddEntry(fit, "Fit Function", "l");
    leg->Draw();

    // Draw reference line and annotation arrow
    TLine *l = new TLine(0, 20, 15, 20);   // Horizontal line at y=20
    l->SetLineWidth(2);
    l->SetLineColor(kOrange);
    l->Draw();

    //Arrow pointing to a specific histogram bin
    double x0 = 6.3;
    int bin = hist->FindBin(x0);
    double y0 = hist->GetBinContent(bin);
    TArrow *arr = new TArrow(10, 30, x0, y0);
    arr->SetLineWidth(2);
    arr->SetArrowSize(0.02);
    arr->SetLineColor(kBlue);
    arr->Draw();

    //Text label near the arrow
    TLatex *t = new TLatex(10, 30, "Important");
    t->Draw();


    //Extract fit parameters and compute mean/sigma ratio
    double mean  = fit->GetParameter(1);
    double sigma = fit->GetParameter(2);
    cout << mean / sigma << endl;   // Print ratio to console
}
```

2. Analyzing the J/ψ resonance

Listing 2. extract2.c

```
void extract2(const char* infile = "simulated_mass.txt") {
    // --- Settings ---
    const int nBins = 100;
    const double minMass = 2.0;   // GeV/c^2
    const double maxMass = 4.0;   // GeV/c^2

    // --- Histogram ---
    TH1F *hMass = new TH1F("hMass", ";Invariant␣Mass␣M_{#mu#mu}␣(GeV/
        c^{2});Entries␣/␣bin", nBins, minMass, maxMass);
    hMass->SetMarkerStyle(20);
    hMass->SetMarkerSize(1.0);

    // --- Random data simulation (replace later with real data
        reading) ---
    TRandom3 rand(42);
    std::ofstream fout(infile);
    const double signalYield = 500;
    const double bkgYield = 2000;

    // Signal: Gaussian centered at 3.1 GeV/c^2 (J/psi mass)
    for (int i = 0; i < signalYield; i++) {
        double mass = rand.Gaus(3.097, 0.05); // mean, sigma
        fout << mass << "\n";
    }
    // Background: exponential falloff
    for (int i = 0; i < bkgYield; i++) {
        double mass = minMass - log(rand.Uniform()) * 0.8; // lambda
        if (mass < maxMass) fout << mass << "\n";
    }
    fout.close();

    // --- Read data ---
    std::ifstream fin(infile);
    double val;
    while (fin >> val) {
        if (val >= minMass && val <= maxMass) hMass->Fill(val);
    }
    fin.close();

    // --- Fit function: Gaussian + exponential background ---
    TF1 *fitFunc = new TF1("fitFunc", "[0]*exp([1]*x)␣+␣[2]*exp
        (-0.5*((x-[3])/[4])**2)", minMass, maxMass);
    fitFunc->SetParameters(100, -1, 200, 3.1, 0.05); // initial
        guesses
    fitFunc->SetParNames("BkgNorm", "BkgSlope", "SigNorm", "SigMean",
         "SigSigma");

    // --- Canvas ---
    TCanvas *c1 = new TCanvas("c1", "STAR-style␣Analysis", 800, 600);
    gStyle->SetOptFit(1111);
    gStyle->SetStatX(0.88);
    gStyle->SetStatY(0.88);

    hMass->Draw("E1");
    hMass->Fit("fitFunc", "R");

```

```
    // --- Legend ---
    TLegend *leg = new TLegend(0.2, 0.55, 0.48, 0.75);
    leg->SetBorderSize(0);
    leg->SetFillStyle(0);
    leg->AddEntry(hMass, "Simulated Data", "lep");
    leg->AddEntry(fitFunc, "Fit: Gauss + Exp", "l");
    leg->Draw();

    // --- Annotation ---
    TLatex text;
    text.SetNDC();
    text.SetTextSize(0.04);
    text.DrawLatex(0.15, 0.85, "#bf{STAR Simulation}");
    text.DrawLatex(0.15, 0.80, "p+p #sqrt{s} = 200 GeV");

    // --- Save outputs ---
    c1->SaveAs("mass_spectrum.pdf");
    c1->SaveAs("mass_spectrum.png");

    // --- Save histogram and fit to ROOT file ---
    TFile outFile("analysis_results.root", "RECREATE");
    hMass->Write();
    fitFunc->Write();
    outFile.Close();
}
```

3. Importing and Analyzing Data from multiple files

Listing 3. multiplefilesupgrade.c

```
#include <TRandom3.h>
#include <TStyle.h>
#include <TLegend.h>
#include <THStack.h>
#include <TH1F.h>
#include <TF1.h>
#include <TCanvas.h>
#include <TFile.h>
#include <iostream>
#include <fstream>

void multiplefilesupgrade() {
    gStyle->SetOptStat(0);

    const int nHists = 6;            // number of histograms per file
    const int nFiles = 10;           // total number of files
    const int nEventsPerFile = 1000; // events per file

    TRandom3 randGen(0); // random seed for reproducibility

    // Parameters for each dataset's Gaussian (mean, sigma)
    double means[nHists]  = {0.0, 0.5, -0.5, 1.0, -1.0, 0.0};
    double sigmas[nHists] = {1.0, 0.8, 1.2, 0.6, 1.5, 2.0};

    TH1F *hist[nHists];
    THStack *hstack = new THStack("stack", "Momentum␣Distributions;P␣
        [GeV/c];Entries");
    TLegend *leg = new TLegend(0.65, 0.5, 0.88, 0.88);
    leg->SetBorderSize(0);
    leg->SetFillStyle(0);

    // --- Create histograms ---
    for (Int_t i = 0; i < nHists; i++) {
        TString name = Form("hist%d", i);
        hist[i] = new TH1F(name, "", 100, -5, 5);
        hist[i]->SetMarkerStyle(20 + i);
        hist[i]->SetMarkerSize(0.8);
        hist[i]->SetMarkerColor(i + 1);
        hist[i]->SetLineColor(i + 1);
        hstack->Add(hist[i], "E1");
        leg->AddEntry(hist[i], Form("Dataset␣%d", i + 1), "lep");
    }

    // --- Generate + Read data ---
    for (Int_t iFile = 0; iFile < nFiles; iFile++) {
        TString filename = Form("input%d", iFile);

        // Generate synthetic data with different means/sigmas per
            histogram
        {
            std::ofstream outfile(filename.Data());
            if (!outfile.is_open()) {
                std::cerr << "Error:␣cannot␣write␣" << filename <<
                    std::endl;
                continue;
            }
```

```
54
55              for (int ev = 0; ev < nEventsPerFile; ev++) {
56                  for (int j = 0; j < nHists; j++) {
57                      double val = randGen.Gaus(means[j], sigmas[j]);
58                      outfile << val << " ";
59                  }
60                  outfile << "\n";
61              }
62              outfile.close();
63          }
64
65          // Read data back into histograms
66          std::ifstream infile(filename.Data());
67          if (!infile.is_open()) {
68              std::cerr << "Warning: could not open " << filename << "
                    for reading." << std::endl;
69              continue;
70          }
71
72          while (true) {
73              for (Int_t j = 0; j < nHists; j++) {
74                  Double_t val;
75                  infile >> val;
76                  if (!infile) break;
77                  hist[j]->Fill(val);
78              }
79              if (infile.eof()) break;
80          }
81          infile.close();
82      }
83
84      // --- Fit each histogram & print parameters ---
85      std::cout << "\n===== Gaussian Fit Results =====" << std::endl;
86      TF1 *fitFunc[nHists];
87      for (Int_t i = 0; i < nHists; i++) {
88          fitFunc[i] = new TF1(Form("fit%d", i), "gaus", -5, 5);
89          fitFunc[i]->SetLineColor(hist[i]->GetLineColor());
90          hist[i]->Fit(fitFunc[i], "RQ"); // RQ = quiet fit, but
                returns result
91
92          double mean   = fitFunc[i]->GetParameter(1);
93          double sigma = fitFunc[i]->GetParameter(2);
94          double meanErr  = fitFunc[i]->GetParError(1);
95          double sigmaErr = fitFunc[i]->GetParError(2);
96          double chi2ndf  = fitFunc[i]->GetChisquare() / fitFunc[i]->
                GetNDF();
97
98          std::cout << Form("Dataset %d:", i + 1)
99                    << " Mean = " << mean << " ± " << meanErr
100                   << " GeV/c, Sigma = " << sigma << " ± " <<
                          sigmaErr
101                   << " GeV/c, Chi /NDF = " << chi2ndf
102                   << ", Entries = " << hist[i]->GetEntries()
103                   << std::endl;
104     }
105
106     // --- Draw ---
107     TCanvas *c1 = new TCanvas("c1", "Momentum Distributions", 1000,
            800);
108     c1->SetGrid();
109     c1->SetTickx();
```

```
    c1->SetTicky();
    hstack->Draw("nostack E1");
    for (int i = 0; i < nHists; i++) {
        fitFunc[i]->Draw("same");
    }
    hstack->GetXaxis()->SetTitleSize(0.045);
    hstack->GetYaxis()->SetTitleSize(0.045);
    hstack->GetXaxis()->SetLabelSize(0.04);
    hstack->GetYaxis()->SetLabelSize(0.04);
    leg->Draw();

    // --- Save ---
    c1->SaveAs("momentum_distributions_with_fits.pdf");
    c1->SaveAs("momentum_distributions_with_fits.png");

    TFile outFile("momentum_distributions_with_fits.root", "RECREATE");
    for (Int_t i = 0; i < nHists; i++) {
        hist[i]->Write();
        fitFunc[i]->Write();
    }
    outFile.Close();

    std::cout << "\nAnalysis complete. Fits overlaid, parameters printed above, ROOT & plots saved." << std::endl;
}
```

4. Analyzing the ρ meson resonance

Listing 4. macro3.c

```
void macro3() {
    gStyle->SetOptStat(0);

    // --- Step 1: Generate simulated signal & background separately ---
    int nSignal = 10000;   // 10% of events
    int nBackground = 90000; // 90% of events

    TRandom3 rand(0);

    // Histograms
    TH1F* histSignal = new TH1F("histSignal", "Signal (Breit-Wigner);Mass [GeV/c^{2}];Entries", 100, 0, 2);
    TH1F* histBackground = new TH1F("histBackground", "Background (Gaussian);Mass [GeV/c^{2}];Entries", 100, 0, 2);

    // Fill signal dataset (Breit-Wigner)
    for (int i = 0; i < nSignal; ++i) {
        double mass = rand.BreitWigner(0.77, 0.15);
        histSignal->Fill(mass);
    }

    // Fill background dataset (Gaussian)
    for (int i = 0; i < nBackground; ++i) {
        double mass = rand.Gaus(0.5, 0.2);
        histBackground->Fill(mass);
    }

    // --- Step 2: Apply realistic error bars (stat + syst) ---
    for (int i = 1; i <= histSignal->GetNbinsX(); ++i) {
        double N = histSignal->GetBinContent(i);
        double statErr = sqrt(N);
        double systErr = 0.05 * N; // 5% systematic
        histSignal->SetBinError(i, sqrt(statErr*statErr + systErr*systErr));
    }
    for (int i = 1; i <= histBackground->GetNbinsX(); ++i) {
        double N = histBackground->GetBinContent(i);
        double statErr = sqrt(N);
        double systErr = 0.05 * N; // 5% systematic
        histBackground->SetBinError(i, sqrt(statErr*statErr + systErr*systErr));
    }

    // --- Step 3: Fit signal with Breit-Wigner ---
    TF1* fitBW = new TF1("fitBW", "breitwigner", 0, 2);
    fitBW->SetParameters(2000, 0.77, 0.15);
    histSignal->Fit(fitBW, "RQ");

    // --- Step 4: Fit background with Gaussian ---
    TF1* fitGaus = new TF1("fitGaus", "gaus", 0, 2);
    fitGaus->SetParameters(8000, 0.5, 0.2);
    histBackground->Fit(fitGaus, "RQ");

    cout << "Gaussian Chi2/NDF (background): " << fitGaus->GetChisquare()/fitGaus->GetNDF() << endl;
```

```
    cout << "Breit-Wigner␣Chi2/NDF␣(signal):␣" << fitBW->GetChisquare
        ()/fitBW->GetNDF() << endl;

    // --- Step 5: Plot results ---
    TCanvas* c = new TCanvas("c", "Invariant␣Mass␣Analysis", 900,
        700);

    histBackground->SetMarkerStyle(20);
    histBackground->SetMarkerColor(kBlue);
    histBackground->SetLineColor(kBlue);

    histSignal->SetMarkerStyle(21);
    histSignal->SetMarkerColor(kRed);
    histSignal->SetLineColor(kRed);

    histBackground->Draw("E1");
    histSignal->Draw("E1␣SAME");

    fitGaus->SetLineColor(kBlue+2);
    fitGaus->SetLineWidth(2);
    fitBW->SetLineColor(kRed+2);
    fitBW->SetLineWidth(2);

    TLegend* leg = new TLegend(0.6, 0.7, 0.88, 0.88);
    leg->SetBorderSize(0);
    leg->AddEntry(histBackground, "Background␣(Gaussian)", "lep");
    leg->AddEntry(histSignal, "Signal␣(Breit-Wigner)", "lep");
    leg->AddEntry(fitGaus, "Gaussian␣Fit", "l");
    leg->AddEntry(fitBW, "Breit-Wigner␣Fit", "l");
    leg->Draw();

    // --- Step 6: Save outputs ---
    c->SaveAs("signal_background.pdf");
    c->SaveAs("signal_background.png");

    TFile outFile("analysis_output.root", "RECREATE");
    histSignal->Write();
    histBackground->Write();
    outFile.Close();

    cout << "Analysis␣complete.␣Separate␣signal␣and␣background␣stored
        ." << endl;
}
```

5. Creating a 2D histogram

Listing 5. $two_{dh}istogram.c$

```
void two_d_histogram() {
    // --- Settings ---
    const int nEvents = 1e6;
    const double meanVal = 0.0;
    const double sigmaVal = 1.0;

    // --- Create histograms ---
    TH2F *h2 = new TH2F("h2", ";x␣[cm];y␣[cm]", 200, -3, 3, 200, -3,
        3);

    // Random generator
    TRandom3 rand(10);

    // Fill 2D Gaussian
    for (int i = 0; i < nEvents; i++) {
        double x = rand.Gaus(meanVal, sigmaVal);
        double y = rand.Gaus(meanVal, sigmaVal);
        h2->Fill(x, y);
    }

    // --- Create canvas with pads for projections ---
    TCanvas *c = new TCanvas("c", "STAR-style␣2D␣Gaussian␣with␣
        Projections", 1000, 1000);
    gStyle->SetOptStat(0);
    gStyle->SetPalette(kBird);
    gStyle->SetNumberContours(100);

    // Pads
    TPad *padMain = new TPad("padMain", "Main␣2D", 0.0, 0.0, 0.8,
        0.8);
    TPad *padX    = new TPad("padX", "X␣projection", 0.0, 0.8, 0.8,
        1.0);
    TPad *padY    = new TPad("padY", "Y␣projection", 0.8, 0.0, 1.0,
        0.8);

    padMain->SetRightMargin(0.15);
    padMain->SetLeftMargin(0.15);
    padMain->SetBottomMargin(0.15);
    padX->SetBottomMargin(0.0);
    padX->SetLeftMargin(0.15);
    padX->SetRightMargin(0.15);
    padY->SetTopMargin(0.15);
    padY->SetLeftMargin(0.0);

    padMain->Draw();
    padX->Draw();
    padY->Draw();

    // --- Draw main 2D plot ---
    padMain->cd();
    h2->Draw("COLZ");
    h2->Draw("CONT3␣SAME"); // add contour lines on top

    // --- X projection ---
    padX->cd();
    TH1D *hX = h2->ProjectionX();
```

```
    hX->SetLineColor(kBlue+1);
    hX->SetFillColorAlpha(kBlue, 0.3);
    hX->GetXaxis()->SetLabelSize(0);
    hX->GetYaxis()->SetTitle("Entries");
    hX->Draw();

    // --- Y projection ---
    padY->cd();
    TH1D *hY = h2->ProjectionY();
    hY->SetLineColor(kRed+1);
    hY->SetFillColorAlpha(kRed, 0.3);
    hY->GetYaxis()->SetLabelSize(0);
    hY->Draw("HIST");
    hY->SetMaximum(hY->GetMaximum() * 1.1);

    // --- Output statistics to terminal ---
    double meanX = h2->GetMean(1);
    double meanY = h2->GetMean(2);
    double sigmaX = h2->GetRMS(1);
    double sigmaY = h2->GetRMS(2);

    std::cout << "=====␣STAR-style␣2D␣Gaussian␣Analysis␣=====" << std
        ::endl;
    std::cout << "Number␣of␣entries:␣" << h2->GetEntries() << std::
        endl;
    std::cout << "Mean␣X␣=␣" << meanX << "␣cm" << std::endl;
    std::cout << "Sigma␣X␣=␣" << sigmaX << "␣cm" << std::endl;
    std::cout << "Mean␣Y␣=␣" << meanY << "␣cm" << std::endl;
    std::cout << "Sigma␣Y␣=␣" << sigmaY << "␣cm" << std::endl;

    // --- Save outputs ---
    c->SaveAs("beamspot_2D_with_projections.pdf");
    c->SaveAs("beamspot_2D_with_projections.png");

    TFile outFile("beamspot_2D_with_projections.root", "RECREATE");
    h2->Write();
    hX->Write("projX");
    hY->Write("projY");
    outFile.Close();
}
```

6. Creating a 3D visual

Listing 6. view2_3d.c

```
void view2_3d() {
    gStyle->SetOptStat(0);

    // --- Canvas ---
    TCanvas *c1 = new TCanvas("c1", "STAR-style 3D Geometry", 1200,
        900);

    // --- Geometry Manager ---
    TGeoManager *man = new TGeoManager("STARgeom", "STAR Detector
        Geometry");

    // --- Materials and Mediums ---
    TGeoMaterial *matVacuum = new TGeoMaterial("Vacuum", 0, 0, 0);
    TGeoMedium   *vacuum    = new TGeoMedium("Vacuum", 1, matVacuum);

    TGeoMaterial *matAl = new TGeoMaterial("Aluminum", 26.98, 13,
        2.7);
    TGeoMedium   *alum  = new TGeoMedium("Aluminum", 2, matAl);

    TGeoMaterial *matSi = new TGeoMaterial("Silicon", 28.085, 14,
        2.33);
    TGeoMedium   *silicon = new TGeoMedium("Silicon", 3, matSi);

    // --- World volume ---
    TGeoVolume *top = man->MakeBox("TOP", vacuum, 50, 50, 50);
    man->SetTopVolume(top);

    // --- Barrel detector (tube) ---
    TGeoVolume *barrel = man->MakeTube("TPC", silicon, 20.0, 22.0,
        40.0);
    barrel->SetLineColor(kBlue - 7);
    barrel->SetTransparency(60);
    top->AddNode(barrel, 0);

    // --- Endcap disk ---
    TGeoVolume *endcap = man->MakeTube("Endcap", alum, 0.0, 22.0,
        1.0);
    endcap->SetLineColor(kGreen + 2);
    endcap->SetTransparency(40);
    top->AddNode(endcap, 0, new TGeoTranslation(0, 0, 40.0));
    top->AddNode(endcap, 1, new TGeoTranslation(0, 0, -40.0));

    // --- Additional inner tracker layer ---
    TGeoVolume *innerLayer = man->MakeTube("InnerTracker", silicon,
        5.0, 5.5, 40.0);
    innerLayer->SetLineColor(kOrange + 7);
    innerLayer->SetTransparency(40);
    top->AddNode(innerLayer, 0);

    // --- Close geometry ---
    man->CloseGeometry();

    // --- Draw geometry ---
    top->Draw("ogl");

    // --- Access 3D viewer ---
```

```
    TGLViewer *view = (TGLViewer*)gPad->GetViewer3D();
    if (view) {
        view->SetStyle(TGLRnrCtx::kOutline); // Outline mode for clarity
        view->CurrentCamera().RotateRad(-0.5, 0.5); // nice angle
        view->RequestDraw();
    }

    // --- Add multiple tracks ---
    TRandom3 r(0);
    for (int t = 0; t < 5; t++) {
        TPolyLine3D *track = new TPolyLine3D();
        track->SetLineColor(kRed + (t % 3));
        track->SetLineWidth(2);
        double phi0 = r.Uniform(0, 2 * TMath::Pi());
        double radius = r.Uniform(5.0, 20.0);
        double zpos = -40.0;
        for (int p = 0; p < 100; p++) {
            double angle = phi0 + p * 0.02;
            double x = radius * cos(angle);
            double y = radius * sin(angle);
            double z = zpos + p * (80.0 / 100);
            track->SetPoint(p, x, y, z);
        }
        track->Draw("same");
    }

    // --- Annotation ---
    TLatex label;
    label.SetNDC();
    label.SetTextSize(0.035);
    label.DrawLatex(0.02, 0.96, "#bf{STAR Detector Geometry - Simulation}");
    label.DrawLatex(0.02, 0.92, "TPC + Endcaps + Inner Tracker");

    // --- Legend for materials ---
    TLegend *leg = new TLegend(0.7, 0.7, 0.9, 0.88);
    leg->AddEntry(barrel, "TPC Barrel (Si)", "f");
    leg->AddEntry(endcap, "Endcaps (Al)", "f");
    leg->AddEntry(innerLayer, "Inner Tracker (Si)", "f");
    leg->Draw();

    // --- Save outputs ---
    c1->SaveAs("star_geometry_full.png");
    c1->SaveAs("star_geometry_full.pdf");

    std::cout << "3D STAR-style geometry saved as PNG and PDF.\n";
}
```

7. Proton-Proton collisions

Listing 7. ppcollision.cc

```
#include "Pythia8/Pythia.h"
#include "TFile.h"
#include "TH1F.h"
#include "TH2F.h"
#include "TCanvas.h"
#include "TLegend.h"
#include "TStyle.h"
#include <cmath>

int main() {
    // --- Settings ---
    int nevents = 5000;   // Number of events
    double eCM = 200.0;     // RHIC energy (GeV) for STAR pp collisions

    // --- Initialize Pythia ---
    Pythia8::Pythia pythia;
    pythia.readString("Beams:idA = 2212");
    pythia.readString("Beams:idB = 2212");
    pythia.readString("Beams:eCM = 200."); // STAR pp energy
    //pythia.readString("SoftQCD:inelastic = on");
    pythia.readString("HardQCD:all = on");
    pythia.init();

    // --- ROOT histograms ---
    TH1F *h_pT  = new TH1F("h_pT",  ";p_{T} [GeV/c];Entries", 100, 0, 5);
    TH1F *h_eta = new TH1F("h_eta", ";#eta;Entries", 100, -5, 5);
    TH1F *h_phi = new TH1F("h_phi", ";#phi [rad];Entries", 64, -M_PI, M_PI);
    TH2F *h_pT_eta = new TH2F("h_pT_eta", ";#eta;p_{T} [GeV/c]", 50, -2.5, 2.5, 50, 0, 5);

    h_pT->Sumw2();
    h_eta->Sumw2();
    h_phi->Sumw2();
    h_pT_eta->Sumw2();

    // --- Event loop ---
    for (int i = 0; i < nevents; i++) {
        if (!pythia.next()) continue;

        for (int j = 0; j < pythia.event.size(); j++) {
            if (!pythia.event[j].isFinal()) continue; // Final state only

            double pT  = pythia.event[j].pT();
            double eta = pythia.event[j].eta();
            double phi = pythia.event[j].phi();

            // STAR acceptance cut
            if (fabs(eta) < 1.0 && pT > 0.2) {
                h_pT->Fill(pT);
                h_eta->Fill(eta);
                h_phi->Fill(phi);
            }
```

```
52
53              // Fill 2D histogram without acceptance cut to show full
                    coverage
54              h_pT_eta->Fill(eta, pT);
55          }
56      }
57
58      // --- Save histograms to ROOT file ---
59      TFile outFile("pythia_histograms.root", "RECREATE");
60      h_pT->Write();
61      h_eta->Write();
62      h_phi->Write();
63      h_pT_eta->Write();
64      outFile.Close();
65
66      // --- STAR-style plotting ---
67      gStyle->SetOptStat(0);
68      gStyle->SetTitleFontSize(0.05);
69
70      // pT plot
71      TCanvas *c1 = new TCanvas("c1", "pT␣Distribution", 800, 600);
72      c1->SetLogy();
73      h_pT->SetMarkerStyle(20);
74      h_pT->Draw("E1");
75      c1->SaveAs("pT_distribution.pdf");
76      c1->SaveAs("pT_distribution.png");
77
78      // eta plot
79      TCanvas *c2 = new TCanvas("c2", "Eta␣Distribution", 800, 600);
80      h_eta->SetMarkerStyle(20);
81      h_eta->Draw("E1");
82      c2->SaveAs("eta_distribution.pdf");
83      c2->SaveAs("eta_distribution.png");
84
85      // phi plot
86      TCanvas *c3 = new TCanvas("c3", "Phi␣Distribution", 800, 600);
87      h_phi->SetMarkerStyle(20);
88      h_phi->Draw("E1");
89      c3->SaveAs("phi_distribution.pdf");
90      c3->SaveAs("phi_distribution.png");
91
92      // 2D pT vs eta plot
93      TCanvas *c4 = new TCanvas("c4", "pT␣vs␣Eta", 900, 700);
94      c4->SetRightMargin(0.15);
95      gStyle->SetPalette(kBird);
96      h_pT_eta->Draw("COLZ");
97      c4->SaveAs("pT_vs_eta.pdf");
98      c4->SaveAs("pT_vs_eta.png");
99
100     // --- Print Pythia statistics ---
101     pythia.stat();
102
103     return 0;
104 }
```