# ROOT Analysis in Java
HEP/ATLAS Group

BY: Khushi Taori
Supervisors: Sergei Chekanov
and Peter van Gemmeren

Abstract

Data from Large Hadron Collider experiments at CERN, such as ATLAS, are stored in ROOT files, a C++ based data analysis framework. In order to make analysis easier, a Java program (Jas4pp) was created to read ROOT files. However, this analysis tool cannot read files created with ROOT V6, an important update with features such as ways to change the input/output capability of data. The purpose of this project was to update the Java program to read files written with ROOT V6. We created a file containing commonly used ROOT objects, including histograms and trees. This file was opened using Jas4pp, and all errors were noted down. The Java library supporting this program was examined, and the issue was found to be that a class of the wrong length was created. Updating this Java library will allow for more efficient analysis of HEP data and collaboration with other fields.

Argonne
NATIONAL LABORATORY
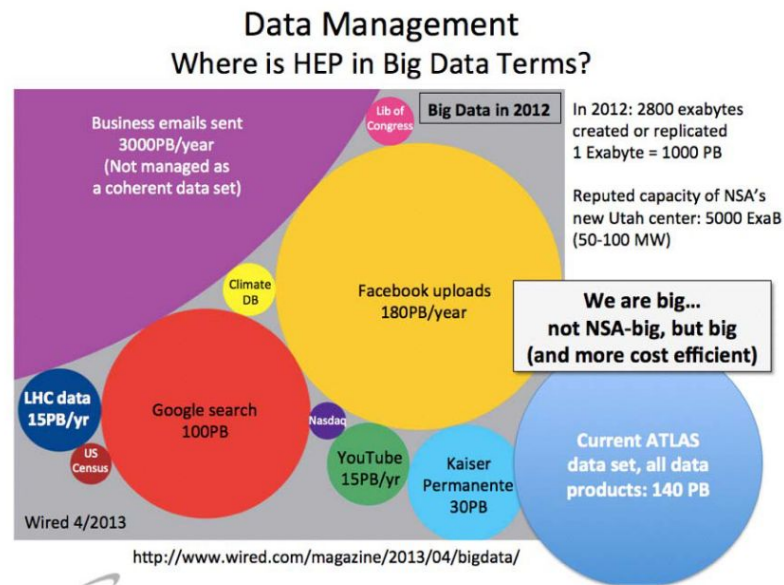
Introduction to the LHC

- Large Hadron Collider (LHC) at CERN

  – Largest particle accelerator in the world

- ATLAS

  – One of two general purpose detectors at the LHC

  – Used for many purposes, such as the search for new particles
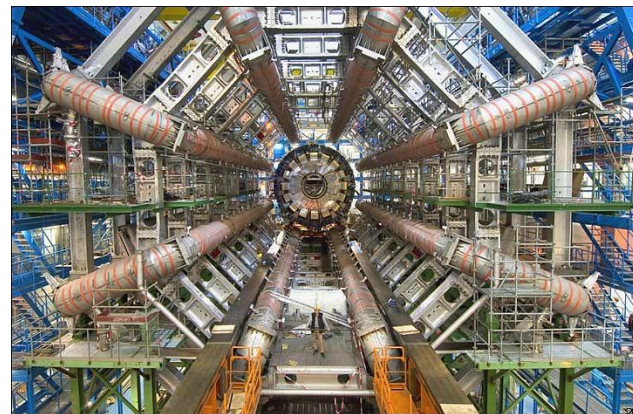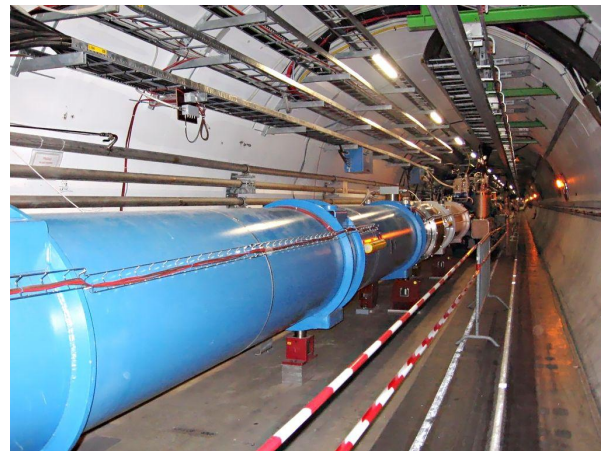
images from home.cern

Data from the ATLAS experiment

- The ATLAS experiment produces a lot of data

- High Energy Physics stores about 1 exabyte of data

- All of this data is processed and written in ROOT format

Data Management
Where is HEP in Big Data Terms?

Big Data in 2012

Business emails sent 3000PB/year (Not managed as a coherent data set)

Lib of Congress

Climate DB

Facebook uploads 180PB/year

In 2012: 2800 exabytes created or replicated 1 Exabyte = 1000 PB

Reputed capacity of NSA's new Utah center: 5000 ExaB (50-100 MW)

We are big... not NSA-big, but big (and more cost efficient)

LHC data 15PB/yr

US Census

Google search 100PB

Nasdaq

YouTube 15PB/yr

Kaiser Permanente 30PB

Current ATLAS data set, all data products: 140 PB

Wired 4/2013

http://www.wired.com/magazine/2013/04/bigdata/

image from www.wired.com/magazine/2013/04/bigdata/

Argonne NATIONAL LABORATORY

What is ROOT?

- C++ based data analysis framework

    – I/O features were designed over 20 years ago

- Problems

    – Can only be run on Linux/Mac

    – Complex with a steep learning curve

    – Analysis of ROOT data is limited to C++ or pyROOT (slow!)

How can we fix these issues with ROOT?

- Create a Java program!

    - More widely used and simpler than C++

    - Faster, easier, and more efficient analysis

    - Lessen the learning curve

    - Allow for collaboration with other fields

    - Can be downloaded on multiple platforms, not just Linux/Mac

    - Support for other scripting languages, can be faster than pyROOT

Argonne
NATIONAL LABORATORY

Jas4pp - Java Analysis Studio For Particle Physics

- Easy installation on many computational platforms (such as Windows 11)

- Easy maintenance on end-user OS

- Can extend with 3rd party libraries

- Created in collaboration: ANL + Jas3 (SLAC) + Hall-B libraries (JLab)
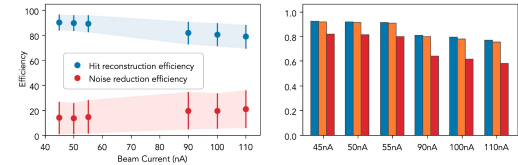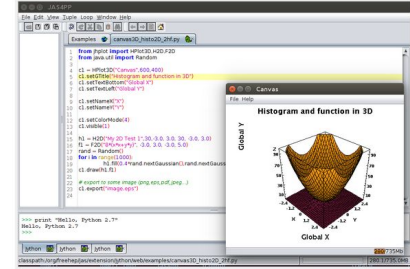
S.Chekanov (ANL), N.A.Graf (SLAC), G.Gavalian (JLAB)   Computer Physics Comm, 262 (2021) 107857
https://arxiv.org/abs/2011.05329
Talk at: APS April 17–20, 2021 Session T19: HEP Data Analysis

Argonne
NATIONAL LABORATORY

Jas4pp - Java Analysis Studio For Particle Physics

- Includes more than just ROOT, with features such as

  - multiple programming languages, I/O libraries, interactive fits, limit settings

  - Physics libraries: event shapes, jet algorithms

  - Tools for Geant4 simulations (tracks, hits, clusters)

  - Universal event display

  - **RootIO java library (*.jar) is included in Jas4pp, but it was not tested for ROOT Versions 5 and 6**

Argonne
NATIONAL LABORATORY

What objects can be opened?

- Using ROOT V5, all objects can be opened without problem in these Java packages. However, this is not the case in V6.

- Created file Example.root using C++

    – Contained many commonly used objects — histograms, graphs, trees

- Documented which objects could and could not be opened using the pre-existing packages

RESULTS

Argonne
NATIONAL LABORATORY

	Visible in Java GUI?	Jython
TH1D	yes	yes
TH2D	yes	yes
TH3D	yes	yes
TProfile	yes	yes
TH1F	yes	yes
TH2F	yes	yes
TH3F	yes	yes
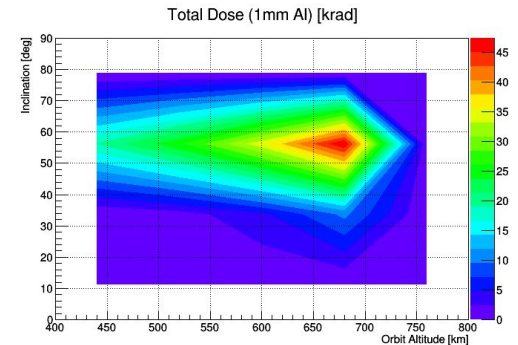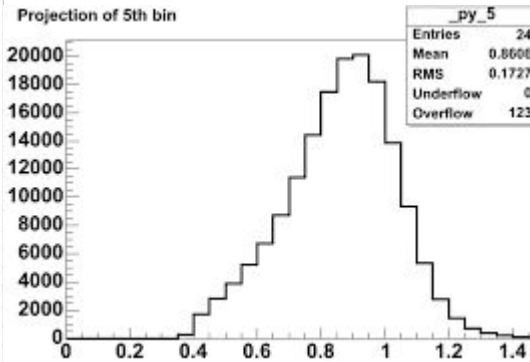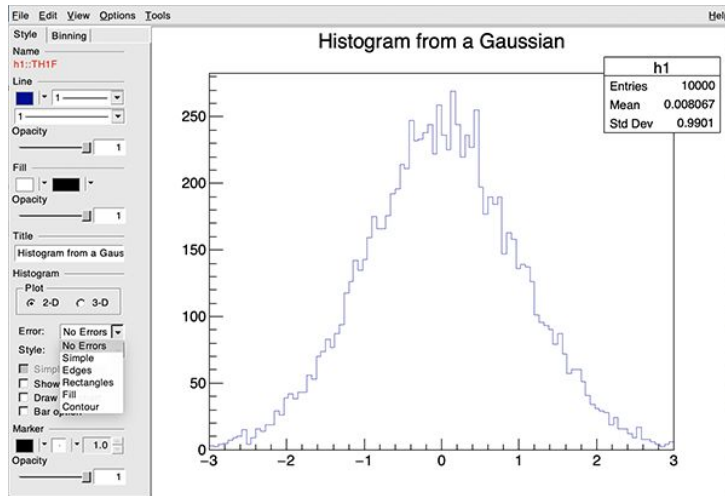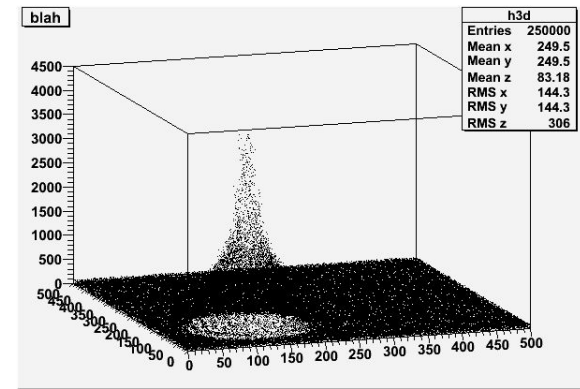
images from root.cern.ch

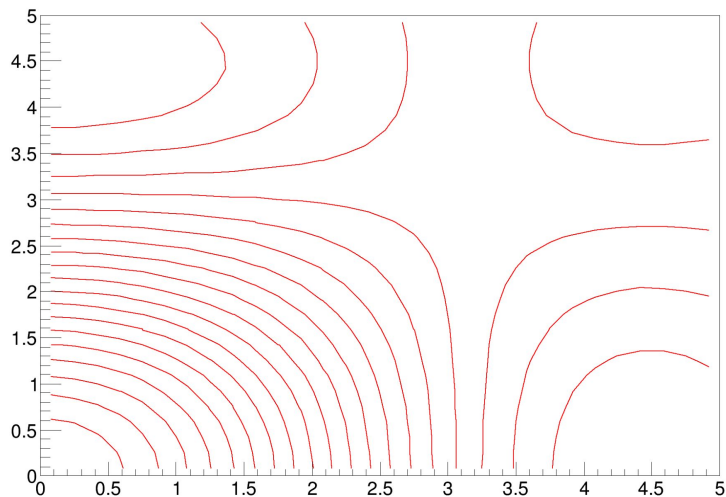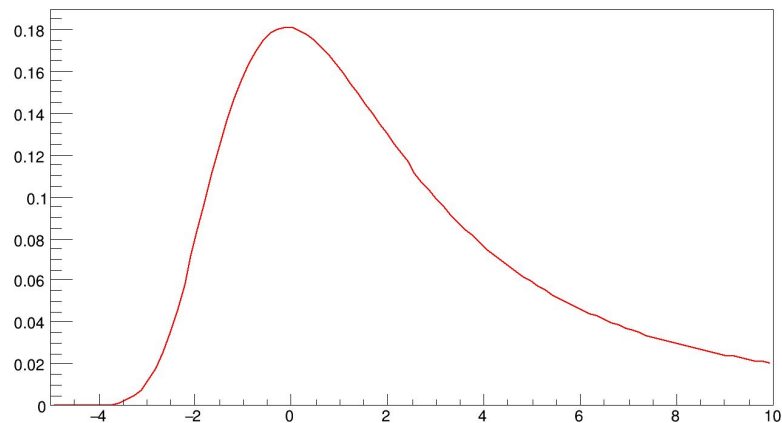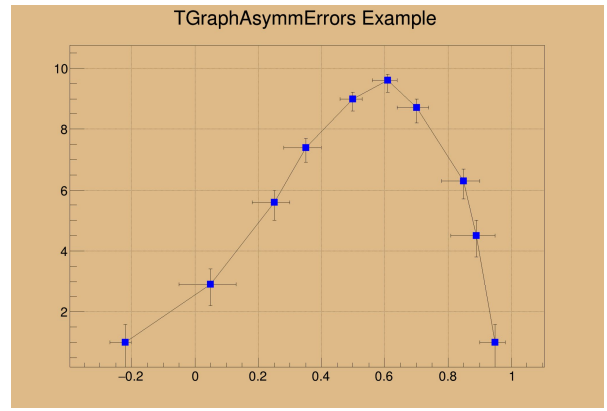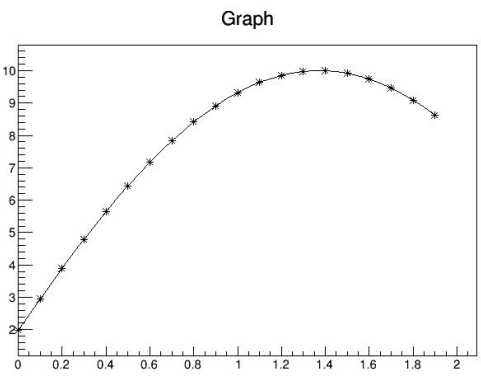|	Visible in Java GUI?	Jython
TF1	java.io.IOException: java.io.IOException: Class not found during object read: TStreamerObjectAnyPointer	
TF2	java.io.IOException: java.io.IOException: Class not found during object read: TStreamerObjectAnyPointer	

sin(x)*sin(y)/(x*y)

TMath::Landau(x,[0],[1],0)

images from root.cern.ch

Argonne
NATIONAL LABORATORY

	Visible in Java GUI?	Jython
TGraph		yes
TGraphErrors		yes
TGraphAsymmErrors		yes

TGraphErrors Example

Graph

TGraphAsymmErrors Example
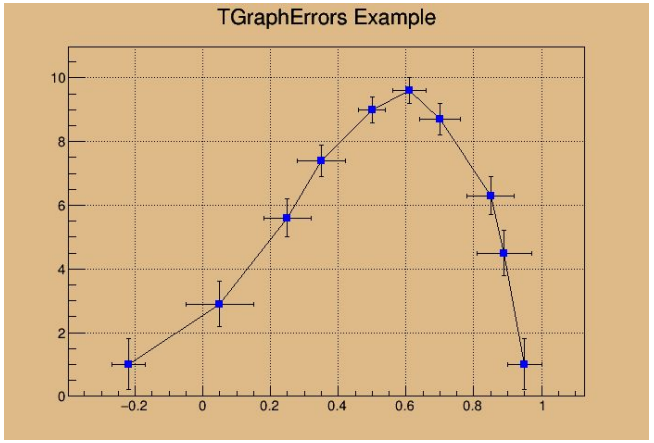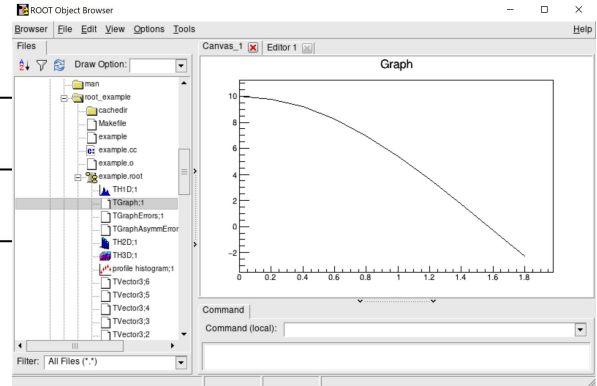
images from root.cern.ch

	Visible in Java GUI?	Jython
TArrayD		Not found in example.root
TClonesArray		Not found in example.root
TLorentzVector		yes
TMap		Not found in example.root
TVector3		yes
TList		Not found in example.root

image from root.cern.ch

Argonne
NATIONAL LABORATORY
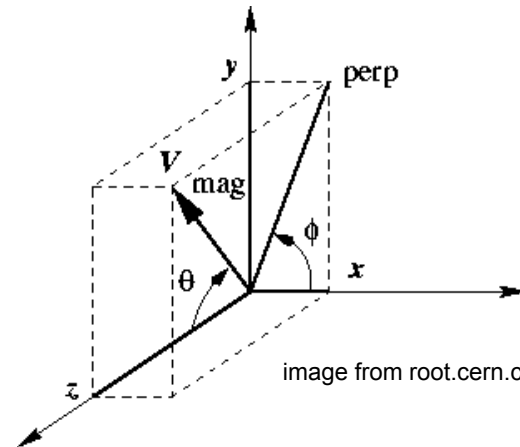
Results

- Many commonly used objects — most notable TTree — could not be opened.
  - Most HEP data is stored in TTrees, making it one of the most important objects
  - Objects such as histograms and graphs are commonly used in analysis
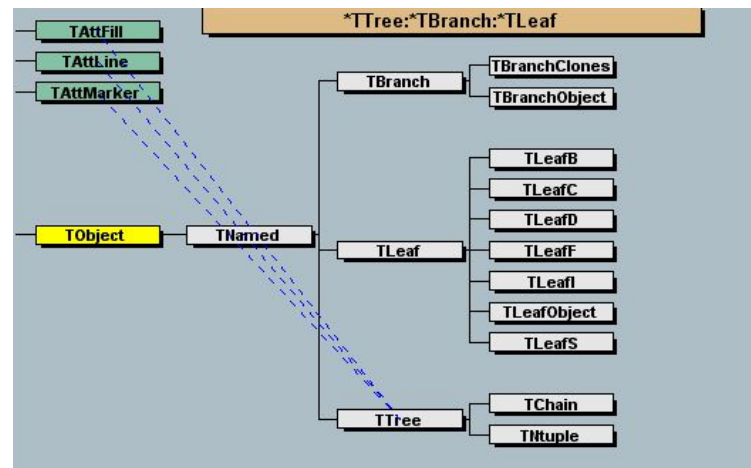
image from root.cern.ch

	Visible in Java GUI?	Jython
TTree		Not found in example.root

Argonne
NATIONAL LABORATORY

Introduction of ROOT Version 6

- ROOT V6 added a lot of changes, including changing from a C interpreter to an In-Time compiler

- Also added TIOFeatures to the TTree class

  - Allows the end-user to change the I/O behavior of data in a TTree

  - This is likely one reason that TTree objects can no longer be opened

Argonne
NATIONAL LABORATORY

Breakdown of RootIOreader

```
┌─────────────────────┐      ┌─────────────────────┐      ┌─────────────────────┐
│ GenericRootClass.java│      │ RootClassLoader.java │      │ RootFileReader object│
│ creates template of  │ ───> │ loads the in-memory  │ ───> │ created by Main.java │
│ in-memory class      │      │ class                │      │ to read the passed   │
│                      │      │                      │      │ in file              │
└─────────────────────┘      └─────────────────────┘      └─────────────────────┘
                                                                      │
                                                                      ▼
┌─────────────────────┐      ┌─────────────────────┐
│ WrongLengthException │      │ TTree object created │
│ called on TIOFeatures│ <─── │ by using             │
│ in-memory class      │      │ RootFileReader object│
└─────────────────────┘      └─────────────────────┘
```
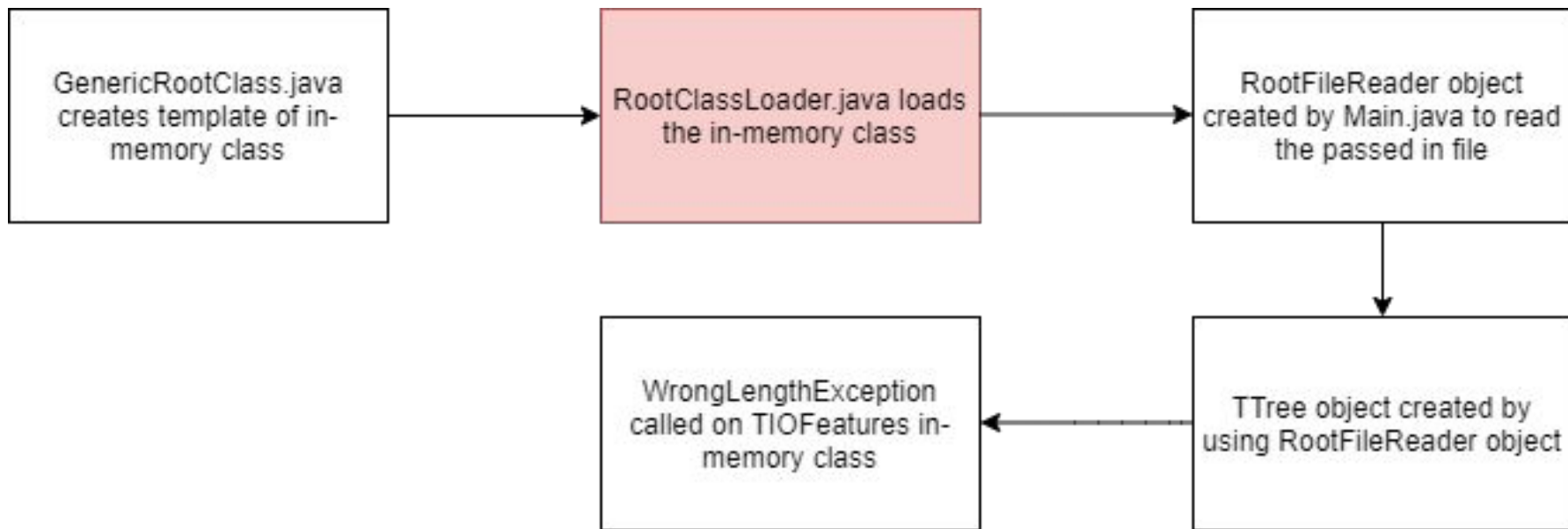
What is the issue?

- TIOFeatures interface is a bit different in structure than some of other interfaces

- TIOFeatures in-memory class is located in a different directory

- Error occurs when RootClassLoader.java is loading in-memory class
  - Creates much larger class than needed, throws a WrongLengthException

- Incorrect TStreamerInfo included for TIOFeatures

Thank you!